BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF MECHANICAL ENGINEERING

FAKULTA STROJNÍHO INŽENÝRSTVÍ

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

DESIGN OF COMMUNICATION INTERFACE IN WBAN SYSTEMS

NÁVRH ŘEŠENÍ KOMUNIKAČNÍHO ROZHRANÍ VE WBAN SYSTÉMECH

MASTER'S THESIS

DIPLOMOVÁ PRÁCE

AUTHOR
AUTOR PRÁCE

Bc. Tomáš Kandera

SUPERVISOR
VEDOUCÍ PRÁCE

doc. Ing. Jiří Krejsa, Ph.D.

BRNO 2020

Specification Master's Thesis

Department:	Institute of Solid Mechanics, Mechatronics and Biomechanics
Student:	**Bc. Tomáš Kandera**
Study programme:	Applied Sciences in Engineering
Study branch:	Mechatronics
Supervisor:	**doc. Ing. Jiří Krejsa, Ph.D.**
Academic year:	2019/20

Pursuant to Act no. 111/1998 concerning universities and the BUT study and examination rules, you have been assigned the following topic by the institute director Master's Thesis:

Design of communication interface in WBAN systems

Concise characteristic of the task:

Bluetooth Low Energy (BLE) technology opens new possibilities in WBAN () applications. Those applications are used for long term monitoring of vital functions and enable bidirectional data transfer in real time. The goal of the thesis is to implement software for the communication of device using proprietary 2.4 GHz protocol together with BLE technology. Design and realization of WBAN device electronics is a part of the thesis. Signal attenuation by human tissue will be experimentally measured and evaluated within the framework of the thesis.

Goals Master's Thesis:

1. Study proprietary protocol and BLE technology and compare both variants.
2. Propose and design demonstrator of circuits taking into account spatial limitations and low energy consumption.
3. Implement software utilizing coexistence of proprietary protocol and BLE technology.
4. Perform experiment to determine signal attenuation by human tissue.
5. Evaluate the solution with respect to energy consumption and transmission quality in various conditions.

Recommended bibliography:

KHAN J.: Wireless Body Area Network (WBAN) for Medical Applications, 2010, DOI: 10.5772/7598, 2010.

AREFIN, M. et.al.: Wireless Body Area Network: An Overview and Various Applications. Journal of Computer and Communications, 5, 53-64. doi: 10.4236/jcc.2017.57006, 2017.

Deadline for submission Master's Thesis is given by the Schedule of the Academic year 2019/20

In Brno,

L. S.

prof. Ing. Jindřich Petruška, CSc.
Director of the Institute

doc. Ing. Jaroslav Katolický, Ph.D.
FME dean

Bibliographic citation

KANDERA, Tomáš. *Design of communication interface in WBAN systems* [online]. Brno, 2020 [cit. 2020-06-12]. Available from: https://www.vutbr.cz/studenti/zav-prace/detail/124658. Master's thesis. Brno University of Technology, Faculty of mechanical engineering, Institute of solid mechanics, mechatronics and biomechancs. Supervised by doc. Ing, Jiří Krejsa, Ph.D.

Abstract

This thesis provides a technical realization of the WBAN device that supports concurrent operation of two wireless communication protocols. Recherche of available solutions is provided by rating multiple products, according to specified demands. The software part contains custom BLE service design, Timeslot API setup, and configuration of peripherals commonly used in WBAN devices. Printed circuit boards for the multiprotocol device prototype and its dedicated controller are made, including custom antenna, power supply management solution, and nRF52810 SoC circuitry. Prototypes testing is summarized in the last chapter with signal attenuation experiment.

Keywords

Multiprotocol, Bluetooth Low Energy, nRF52810, Timeslot API

Abstrakt

Táto diplomová práca sa zaoberá technickou realizáciou WBAN zariadenia, ktoré je schopné súbežne komunikovať prostredníctvom dvoch bezdrôtových komunikačných protokolov. Rešerš dostupných riešení je dostupná s hodnotením jednotlivých produktov podľa vopred stanovených kritérií. Návrh z hľadiska software obsahuje tvorbu vlastného BLE servisu, konfiguráciu Timeslot API a periférií často využívaných vo WBAN zariadeniach. Výroba dosiek plošných spojov pre zariadenie a dedikovaný ovládač zahŕňa vlastnú anténu, nabíjací obvod batérie a nRF52810 čip. Testovanie prototypov je zhrnuté v poslednej kapitole spoločne s experimentálnym meraním útlmu živočíšneho tkaniva.

Kľúčové slová

Multiprotokol, Bluetooth Low Energy, nRF52810, Timeslot API

Rozšírený abstrakt

Táto práca sa zaoberá tvorbou zariadenia, ktoré je svojim uplatnením vhodné pre použitie v sietiach v blízkosti ľudského tela, známych pod pojmom Wireless Body Area Network (WBAN). V čase písania tejto práce sú WBAN zariadenia na vzostupe najmä v medicínskych aplikáciách súvisiacich s dlhodobým monitorovaním pacientov v rámci ich bežných životov. Jeden z dôvodov, ktorý umožnil tento trend je uvedenie technológie Bluetooth Low Energy (BLE) na trh. Bluetooth ako komunikačný protokol je známy po mnohé roky a jeho hlavnou výhodou je adaptácia v drvivej väčšine súčasných smartfónov. Možnosť posielania veľkých súborov, uskutočňovanie telefonických hovorov prostredníctvom handsfree zariadení, atd. má za následok vyššiu energetickú náročnosť v porovnaní s inými protokolmi. Nízkoenergetická verzia BLE predstavuje optimalizáciu vďaka ktorej môžu byť zariadenia napájané batériami s nízkou kapacitou a napriek tomu vydrží byť v prevádzke niekoľko mesiacov (v závislosti na množstve prenesených dát a ďalších faktoroch). Spätná kompatibilita umožňuje komunikáciu takýchto zariadení s väčšinou súčasných smartfónov, čo prináša nové možnosti pre rôzne typy aplikácií spojené s ľudským telom. Myšlienka WBAN sietí je sofistikovanejšia ako inteligentné hodinky prepojené s mobilným telefónom. Je to spôsob ako získať a spracovať informácie o zdravotnom stave pacienta bez nutnosti osobnej návštevy lekára. Dáta zo senzorov sú prostredníctvom BLE prenesené do smartfónu, odkiaľ môžu byť prostredníctvom cloudových služieb uložené do databázy pacienta, kde budú dostupné konkrétnemu lekárovi. Jednými z prvých komerčne dostupných produktov s týmto typom technológie sú BLE kardiostimulátory a automatizované dávkovače inzulínu u pacientov s cukrovkou.

Napriek výhodám BLE technológie môže vzniknúť požiadavka, aby bolo zariadenie schopné komunikovať prostredníctvom dvoch protokolov. Možné situácie v ktorých tento problém môže nastať sú rozobrané v texte práce.

Účelom tejto práce je návrh zariadenia súbežne komunikujúceho prostredníctvom BLE a proprietárneho protokolu poskytnutého výrobcom čipu. V praxi to zahŕňa tvorbu obojsmernej komunikácie so smartfónom a dedikovaným ovládačom, pričom je pri tvorbe softvéru kladený dôraz na univerzálnosť v prípade použitia v ďalších aplikáciách. V prvej kapitole práce sú vypísané požiadavky pre všetky spomenuté zariadenia a ich periférie. Druhá kapitola zahŕňa problematiku a terminológiu spojenú s bezdrôtovou komunikáciou. Nasleduje výber vhodného hardvéru podľa zadaných kritérií, pričom ako najvhodnejší kandidát sa ukázal produkt nRF52810 od spoločnosti Nordic Semiconductor. Toto riešenie je kompromisom z hľadiska vysielacieho výkonu, spotreby, pamäte a technickej podpory. Tvorba softvéru bola vykonaná za pomoci vývojovej dosky nRF52 DK, pomocou ktorej je možné emulovať zvolený hardvér.

Samotná tvorba softvéru je zhrnutá v kapitole 4. Ako prvá je popísaná konfigurácia jednotlivých periférií. Zariadenie meria vstupné napätie pomocou ADC prevodníka, nameraná hodnota je sprostredkovaná do zvyšných dvoch zariadení. Na strane dedikovaného ovládača je doručená informácia o stave nabitia zobrazená prostredníctvom LED diód, signalizácia je riadená pomocou nameraného vybíjacieho profilu použitej batérie. Ovládač na sebe obsahuje 2 tlačidlá, pomocou ktorých je možné meniť mód v ovládanom zariadení. Každý mód má priradenú funkcionalitu podľa tab. 4.1. Jeden z módov obsahuje okrem LED signalizácie aj ovládanie malého vibračného motora, ktorý slúži ako predpríprava na haptickú spätnú väzbu, ktorá môže byť použitá pri indikácii poruchy, akou je vybitie batérie a pod. Motor je ovládaný pomocou PWM modulácie vstupného signálu.

Nasleduje konfigurácia potrebná pre použitý proprietárny protokol Gazell. V tejto časti je riešená problematika možnej interferencie dvoch identických zariadení. Hlavnou výhodou protokolu Gazell je nenáročná inicializácia v porovnaní s robustným BLE.

Aby bolo možné protokol BLE využívať, je nutné inicializovať niekoľko jeho súčastí. Zariadenie v tejto práci ihneď po zapnutí prejde do módu advertising, kedy prostredníctvom vysielania určitého dátového paketu hľadá vhodné spojenie. Takéto zariadenie je bežne pozorovateľné pomocou zoznamu Bluetooth zariadení v smartfóne. BLE pracuje s užívateľskými dátami pomocou hierarchie zloženej z atribútov, pričom sa rozlišujú nasledovné typy atribútov:

- Profile
- Service
- Characteristic

Dva atribúty typu service sú vytvorené vrátane potrebných atribútov typu characteristic, jeden pre meranie napätia batérie a druhý pre funkcionalitu zmeny módov. Spojenie dvoch protokolov je realizované pomocou modulu Timeslot API. Táto knižnica umožňuje selekciu časových úsekov, v ktorých je proprietárnemu protokolu umožnený prístup k perifériám potrebným k vysielaniu dát. Týmto spôsobom dokáže zariadenie reagovať pomocou dvoch protokolov súčasne, a to s minimálnou odozvou.

Piata kapitola sa venuje tvorbe dosky plošných spojov (DPS). Súčasťou je aj výber vhodného zdroja a nabíjacieho obvodu. Vytvorené prototypy slúžia k testovaniu integrácie komunikačného WBAN rozhrania s bezdrôtovým nabíjaním z [29] a v neposlednej rade ako demonštrátor funkcionality softvéru. Použité riešenie antény tiež nebolo možné realizovať na vývojovej doske z dôvodu iného konštrukčného riešenia. Posledná kapitola je venovaná testovaniu sústavy z rôznych hľadísk, uplatniteľných v podmienkach bežnej prevádzky. Počas merania boli všetky užívateľské periférie vypnuté, aby bola interpretácia výsledkov vztiahnutá iba na komunikačné rozhranie. Maximálny vysielací výkon +4 dBm bol nastavený pre BLE aj Gazell. Meranie spotreby ukázalo nepomer medzi jednotlivými protokolmi, hlavne v nepripojenom stave. Protokol Gazell vykazuje násobne väčšie energetické nároky, bolo nutné ošetrenie z hľadiska maximálneho počtu pokusov o spojenie. Časové závislosti prúdu pri rôznych stavoch boli spracované v programe Matlab, kde boli určené aj stredné hodnoty prúdu. Ďalším z testov bolo meranie dosahu vysielaného signálu. V tejto časti bolo hlavnou úlohou porovnanie vyrobených prototypov s vývojovou doskou z hľadiska parametrov antény. Meranie bolo uskutočnené prostredníctvom merania veličiny RSSI pre fixné vzdialenosti v uzavretom priestore Porovnanie ukázalo uspokojivé výsledky, útlm pozorovaný v pracovnom rozsahu by bolo možné minimalizovať presnejším impedančným prispôsobením prototypovej antény.

Posledná časť kapitoly obsahuje experiment, ktorý mal za úlohu vyhodnotiť útlm signálu spôsobený ľudským telom, v prípade možnej aplikácie v implantovanom zariadení. Opäť bolo v tomto prípade využité meranie RSSI. Ľudské tkanivo bolo nahradené živočíšnym, pričom pri úplnom zakrytí DPS bol útlm signálu natoľko významný že nebolo možné nadviazať spojenie medzi dvoma zariadeniami. Riešením bolo použitie povrchovej antény, vďaka ktorej boli výsledky porovnateľné s meraním útlmu v otvorenom priestore.

Zariadenie vytvorené v tejto práci je schopné komunikovať s dvoma ovládačmi používajúcimi dva rôzne protokoly. Vďaka malým rozmerom je možné tento typ zariadenia implementovať do produktov spojených s prevenciou a ochranou zdravia na základe monitorovania životných funkcií. Softvér je pripravený na vývoj front-end BLE aplikácie pre smartfóny.

Declaration

I declare that I have elaborated my master's thesis on the theme „*Design of communication interface in WBAN systems*" independently with the use of technical literature and other sources of information which are all quoted and detailed in the list of literature at the end of the thesis.

Brno ....

Tomáš Kandera

Acknowledgment

I would like to thank doc. Ing. Jiří Krejsa, Ph.D., Ing. Jan Hrbáček, Ph.D., and entire Bender Robotics team for a professional approach and useful bits of advice during the creation of this thesis.

Thanks to my team-mate Ing. Josef Bednařík for an excellent collaboration on the project.

Thank you, my family, for support through all of my studies.

Contents

Introduction

Rapid technical improvement of IoT devices and wireless communication systems boosts their potential in advanced healthcare services. Continuous monitoring with further diagnostics of human body processes, maintenance of chronic conditions, etc. become more accessible thanks to increasing data throughput and minimized energy consumption [1]. Sensor nodes and actuators used for application-specific purposes in close proximity of the human body are grouped as a *Wireless Body Area Network*. The widespread use of smartphones brings new ideas for the WBAN system's versatility enhancement in the field of personal healthcare. The introduction of *Bluetooth Low Energy* technology allows developers to bridge even the most minimalistic WBAN systems with smartphones. This brings new opportunities into medical disciplines that utilize cardio stimulators, insulin pumps, Brain-Computer Interfaces, etc.

The robust architecture of Bluetooth makes such applications possible to make, yet Bluetooth is not the most widespread communication protocol in medical devices despite Bluetooth's approval according to the *Personal Health Devices standard* (ISO/IEEE 11073). Common WBAN device usually communicates with a dedicated host instead of smartphone and developers of such applications tend to use vendor-specific proprietary protocols. These share fundamentals with Bluetooth protocol stack, but they are much more simplified in terms of implementation [2]. To prove the level of simplification, sample application with identical data handling requirements is created by using both proprietary protocol and BLE service.

The main goal of this thesis is to design a solution that concurrently shares data from the WBAN device both into the dedicated radio controller by proprietary protocol and into smartphone by using BLE services. To ensure the versatility of the solution, a bidirectional data flow is allowed so that measured data are obtained, and actuation elements of the device can be managed. The hardware selection process with a custom PCB design is included. The testing procedure of the entire system is presented with power and range examination.

Wireless body area network

WBAN system is not the smartwatch with heart rate monitor and colorful smartphone application. It is a *communication model* that suits the demand for a robust yet compact system with modular architecture for medical disciplines, capable of monitoring rehabilitation, prevention, or diagnostic events. The long-term goal is to make the monitoring process a part of daily activities, not only in hospitals and ambulances [3]. Daily monitoring of vital functions with effective data processing can unveil health issues in less time delay. The global solution would allow simultaneous scanning of multiple patients with reduced personal visits of already busy medical staff.

Fig. 1: WBAN data flow shows a common topology of the WBAN network consisting of sensor nodes around the human body with wireless communication features since cable connections with bulky connectors are one of the most limiting factors for long-term daily use. Data acquired from sensors are transferred to gateway (e.g. smartphone, communication module). Data can be processed or moved to cloud service from where they are accessible for healthcare personnel.

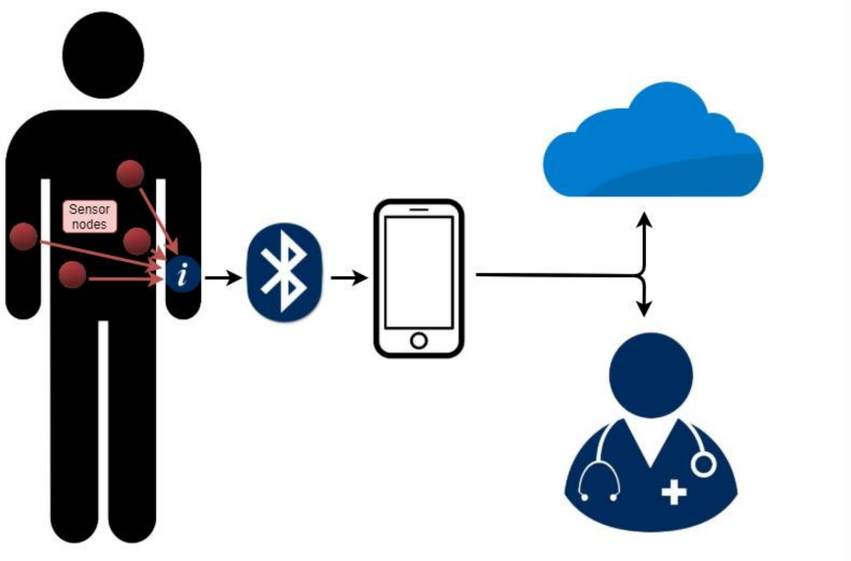

Fig. 1: WBAN data flow

WBAN is a relatively new term in the healthcare environment, yet first successful attempts are made towards its implementation into society. One of the commercial products in this field are *Medtronic's Azure* pacing systems for Bradyarrhythmia management. These are BLE implemented cardio stimulators that connect with the smartphone app. providing notifications and relevant alerts about the pacemaker, in case of atrial fibrillation detection [4]. Bluetooth-equipped insulin pens or glucometers connected with mobile phones are making their way to the healthcare market.

Chapter 1
Problem formulation

The main goal of this thesis is to create a system according to the requirements specified in this chapter. Focus is given on the description and creation of a *multi-protocol* communication interface suitable for personal health purposes. Possible usage in the on-skin and implantable devices requires minimalistic size. To be able to control and present data flow, LED, buttons, and small vibration motor are used. Wireless communication runs between three devices, data exchange principle is based on the following procedures. Functions are created to be open for further extensions that can be easily implemented, depending on the current use-case.

On-body device:

- For the rest of the thesis marked as *Device*[1]
- Small form factor with low energy consumption demand
- Data exchange by BLE and 3rd party protocol
- 4 modes of operation, for the rest of the thesis marked as *WBAN modes*
- Current WBAN mode indicated by LEDs
- Measured battery status concurrently sending to *Host* and *Smartphone*
- Haptic feedback based on the voltage reading

Remote controller:

- For the rest of the thesis marked as *Host*
- Communication with *Device* by 3rd party proprietary protocol
- Access to *Device*'s WBAN mode by button press
- Indication of *Device* battery level by LEDs

Smartphone:

- Access to the *Device* voltage level by BLE service
- Access to *WBAN mode* change in *Device* by BLE service

[1] This is done to avoid confusion with the term device as generic technical piece of work.

Chapter 2
Theoretical overview

Topics that describe the physical aspects of key parts in the wireless system environment are mentioned in this chapter. Wireless communication is based on electromagnetic radiation of radio waves. Information in the form of an AC electrical signal is sent into a radio transmitter. The transmitter generates the carrier wave of a certain radio frequency (RF), used to send the information over the air. Carrier wave is changed over time according to given signal information, the process is called carrier wave modulation. This wave is captured by the antenna on the receiver side. The electromagnetic field of a certain power rating is created. Antenna at the receiver side reacts to the EMF by generating a voltage signal used for decoding the transmitted information.

The rating of transmitter power and receiver sensitivity of devices in the field of radio communication is typically expressed in dBm units. It is a standard power ratio in decibels but referenced to 1 milliwatt. Proper notation of this unit should be dBmW, however dBm became a used term [5]. Equation (2.1) is a conversion between power P_{watt} expressed in Watts and power P_{dBm} expressed in dBm units.

$$P_{dBm} = 10 \log \left(\frac{P_{watt}}{10^{-3}} \right) \tag{2.1}$$

This unit of power is used for its advantage of expressing very big and very small amounts of power within a few digits. When the power of signal is smaller than 1 mW, dBm value is negative and vice versa. Double the P_{WATT} power means a +3 dBm increase. The maximal transmitter power of the BLE device is limited to +10 dBm by regulations necessary for operation in the reserved frequency band. Manufacturers set maximal power value individually, typically 0 dBm, +4 dBm, or +8 dBm, depending on energy consumption requirements. On the other hand, receiver sensitivity is a value for which the signal can be processed with no more than a given bit error rate. Typical sensitivity is usually close to the -90 dBm.

In wireless communication, there are multiple methods on how to measure the quality of a received signal. This can be helpful in the optimization of transmitting power that links to energy consumption. Two main quality ratings are common for Bluetooth applications, RX and RSSI. RX method is used to measure received power in mW or dBm units, RSSI is just a value from a predefined scale with no physical unit. RSSI scale is defined by chip manufacturer and there is no mandatory relationship between physical RX and RSSI. This is the reason why two different vendor devices on the same fixed distance can measure different RSSI values from a single transmitter [6].

2.1 Antenna

Three mechanical variations of antennas are common for BLE devices: microstrip antenna, chip antenna, and wire antenna. They are further divided into monopole and dipole antennas. The right choice usually depends on space constraints [7].

Common parameters that specify antenna's properties are its frequency, gain, impedance, and length. Typically, 50 Ω monopole antennas with omnidirectional radiation pattern and the resonant frequency of around 2.45 GHz are used for BLE applications. Antenna impedance consists of capacitance and inductance components. Antenna tuning means matching its impedance with given system impedance (50 Ω is typical impedance used in 2.4 GHz devices). Proper connection link design for the active part of the antenna is necessary since it can be the reason of limited performance. SoC vendors usually include guidelines for proper antenna design compliant with their solution. Useful resources about antenna design process are available at [7], [8], [9].

Fig. 2.1: Monopole antenna [10]

2.2 Smith charts

Matching the antenna resonant frequency with operating frequency is useful for the proper functionality of wireless communication. Smith chart is a circular graph of complex reflection overlaid with an impedance and admittance grid [11]. It is a system of circles and arcs along which are the observed values constant. The central point of the circle represents system impedance; antenna impedance is located somewhere on the graph. Matching means moving the antenna point as close to the center as possible. This can be done by modifying antenna design, in case of PCB antenna, by changing its length. When antenna modification is not possible, adding passive components (capacitors and inductors) is used instead. A network composed of a series inductor and shunt capacitor is connected between the physical antenna and antenna pin of the SoC.

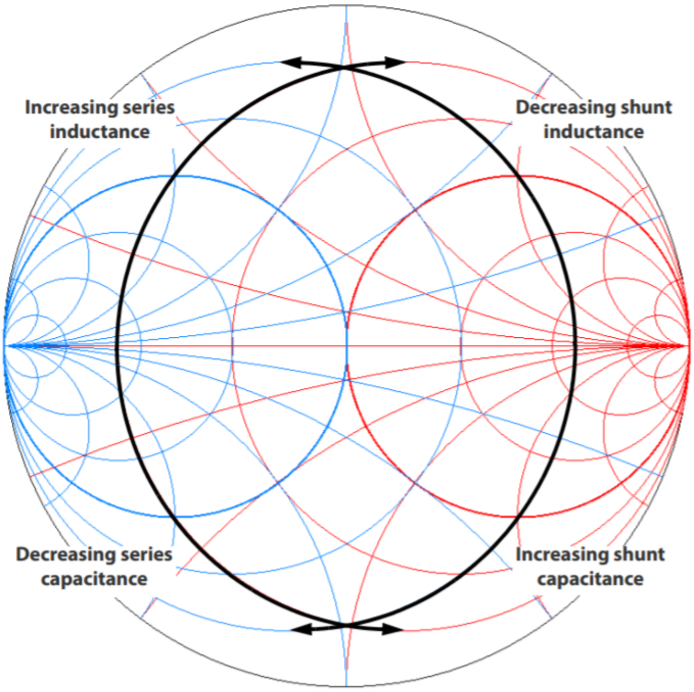

Fig. 2.2: Impedance matching using Smith chart [11]

2.3 Human body interaction

Output power transmitted from the antenna is limited due to human exposure by international guidelines. Therefore, every gadget must be tested by an internationally agreed method so that it can be considered as safe for use. Devices that use UHF frequencies are evaluated by a method called *Specific Absorption Rate*. SAR is a part of the CE certification mark that indicates conformity with health, safety, and environmental protection standards for products sold within the *European Economic Area* [12].

WBAN nodes are expected to be used either as non-invasive body surface devices or in-body implants. Surface devices cause fewer problems in terms of communication if at least one of the antennas (*Device* or *Host*) is not fully covered in human tissue. In case of an implantable *Device*, human body attenuation of transmitted waves becomes a significant aspect. Attenuation is caused by water absorption of the emitted RF signal energy. The human body contains a significant amount of water, degradation of signals at UHF frequencies is negligible. Path loss models specify, how the signal emitted from the transmitter is attenuated by various layers of the human body before it reaches the receiver [13]. This phenomenon can be observed even with a simple experiment when the smartphone antenna covered in hands breaks down the communication link. One solution can be increased transmitter power, although this parameter is limited by CE regulations and battery life. Bluetooth signals cannot be transmitted directly from an invasive body implant. The following layout designs of implant devices are considered to solve this problem.

The implantable device with an exposed antenna places the sensor nodes inside the body with electronics either implanted or not. The antenna is located on the skin as illustrated in Fig.2.3: Antenna over the skin. Wire antenna is suitable for this use case if made with aspects of biocompatibility. However, a wired connection between the sensor node and electronic device limits this solution because measured signals with low voltage levels can be corrupted by noise caused by the transmission line. Wires inside the body can also cause limited mobility.

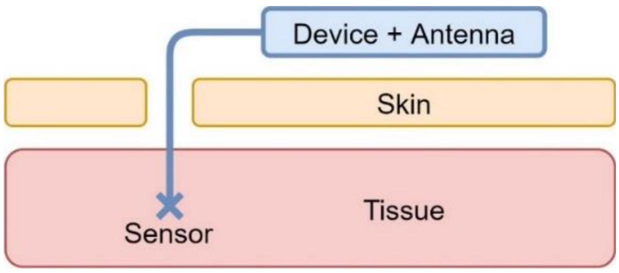

Fig.2.3: Antenna over the skin

Another variant is a device with two modules where one is implanted and second is placed outside the body. The implanted module acquires the data and transfers them to the external module wirelessly. Magnetic coupling is a used method for this type of communication channel, that is feasible in human tissue. This solution minimizes the use of wire connections because both power and information are supplied wirelessly by encoding the data as modulation of power carrier in case of the forward flow of data [14]. The external module processes the data received from the implant and broadcasts them by Bluetooth link.

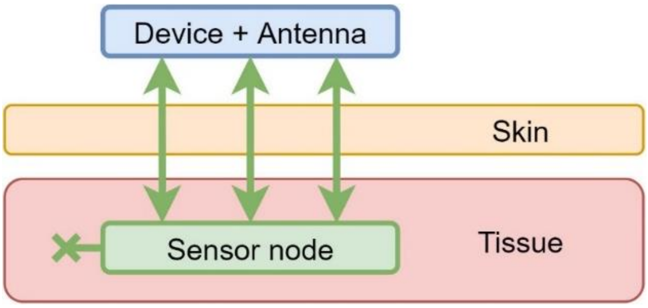

Fig. 2.4: Two modules

2.4 Haptic feedback

Haptic feedback provides a means of communication with the user. Small DC motor with an eccentric rotating mass in connection with the human body produces vibrations. Vibration feedback in WBAN application urges warning signals either related to the user's health state or hardware problem. The most common hardware issue of the WBAN *Device* is discharged onboard battery. Applications where an uninterrupted operation is required demand the user's attention when the state of charge is low. Haptic feedback can be used as the last stage of indication when the user does not react to other signals, e.g. smartphone notifications or LED indication.

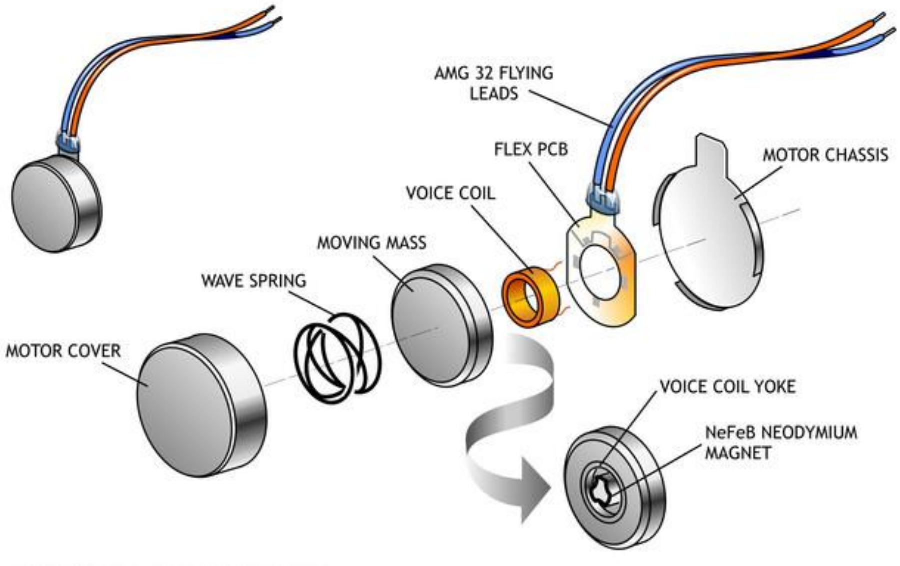

Fig. 2.5: Vibration motor for haptic feedback [15]

Customizable vibration profile useful for optimizing the haptic feedback from aspects of sensitivity and power management. *Pulse width modulation* is used to control the average value of voltage by changing the duty cycle of the signal.

2.5 ADC conversion

Successive-approximation-register analog-to-digital converters[2] are commonly used solutions in embedded systems with decent precision under 5 Msps. SAADC is composed of the comparator, successive-approximation register logic, and digital-to-analog converter. Implemented binary search algorithm tests all bits from MSB to LSB to get a binary number that is equal to decimal input. During the process, the temporary result is compared with the input signal. This feedback adjusts bits to create a precise result of the conversion.

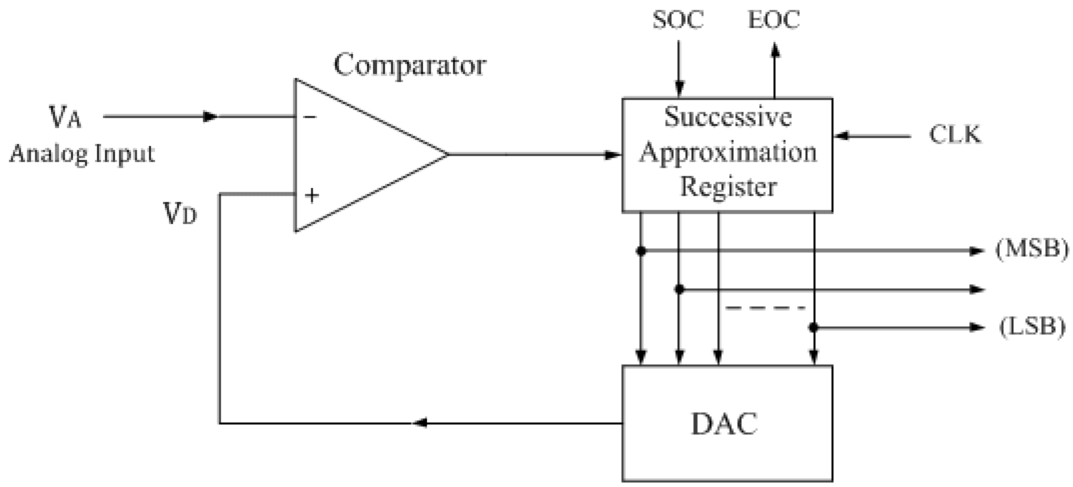

Fig 2.6.: SAR ADC working principle

2.6 Wireless communication protocols

Bluetooth and proprietary protocols described in this chapter operate at 2.4 GHz ISM (Industrial, Scientific, and Medical) frequency band. ISM means a reserved part of the frequency spectrum, all included frequencies are listed in [16]. Devices working in these bands are free from license fees, that is why short-range systems in this area of frequencies are so common. A preferred usage of 2.4 GHz band amongst the others is also because of better performance retaining the same antenna size requirements.

2.7 Bluetooth Low Energy

Bluetooth is a robust communication standard with an extensive number of features available in almost every commercial electronic device. It is capable of file transfer, real-time phone calls (handsfree), etc. On the other hand, Bluetooth Bluetooth Low Energy is

---

[2] SAADC acronym is used for the rest of the thesis

specially developed for low-power critical applications like battery-operated IoT devices. Optimization procedures had to be made in every layer of the Bluetooth stack. The Ultra-low-power feature of this protocol is the reason why it is attractive for WBAN applications. Mentioning all BLE properties takes thousands of pages (Bluetooth Core documentation is 3000+ pages). The following sections describe the most frequent terms used in the BLE environment and the features necessary for understanding the practical part's working principle.

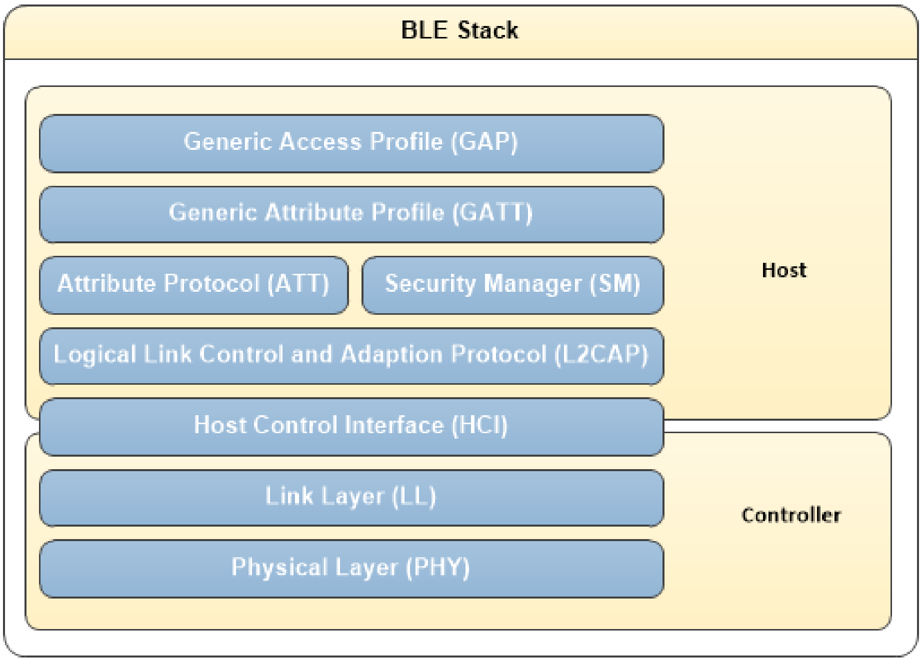

Fig. 2.7: BLE protocol stack

2.7.1 Physical Layer

The main task of a physical layer is electromagnetic radiation handling with the use of a hardware SoC with integrated radio[3]. Bluetooth uses short-wavelength UHF radio waves in the ISM radio band ranging from 2.400 to 2.4835 GHz. 40 radio channels are utilized with 2 MHz spacing. Two types of channels are recognized: advertising and data transfer. Advertising channels are used for device discovery and connection establishment. Data channels form bidirectional communication between two devices.

To prevent interference with other devices working at the ISM band, a frequency hopping feature is used. This means that channel used for data transfer is intentionally changed over with time so that possible interference only affects a small part of transmitted data. Adaptive frequency hopping used in BLE finds the most jammed channels (e.g. by Wi-Fi or Microwave) at first by scanning energy content, then avoids them by selecting more available channels.

[3] Radio is a term that merges all hardware peripherals used for wireless data transfer

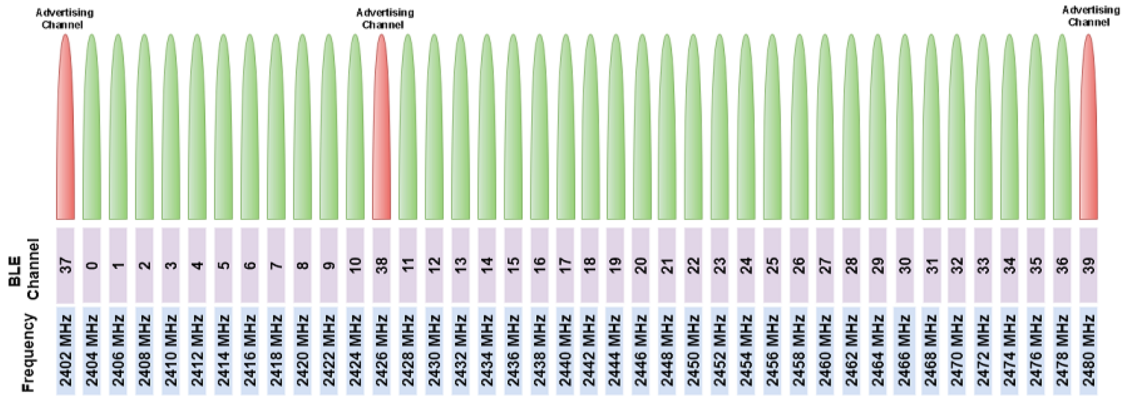

Fig. 2.8: Bluetooth channels [17]

2.7.2 Link layer

The Link layer specifies the interaction between two devices, which includes advertising, scanning and connection events, security, etc. During the advertising event, the peripheral device acts as an advertiser. The central device acts as a scanner and it is the one that starts a connection. Advertiser broadcasts advertising packets within a specified time interval ranging from 20 ms to 10.24 s. The scanner sends a scan request in case it wants to connect to the advertiser. In response, a scan response packet is sent by the advertiser. After this procedure advertiser becomes slave and scanner becomes master, the connection is now established. Master is responsible for managing the connection and connection parameters control. The slave always sends an acknowledgment packet. In a connected state, data is exchanged via data channels in a format specified by the *Protocol Data Unit* [18].

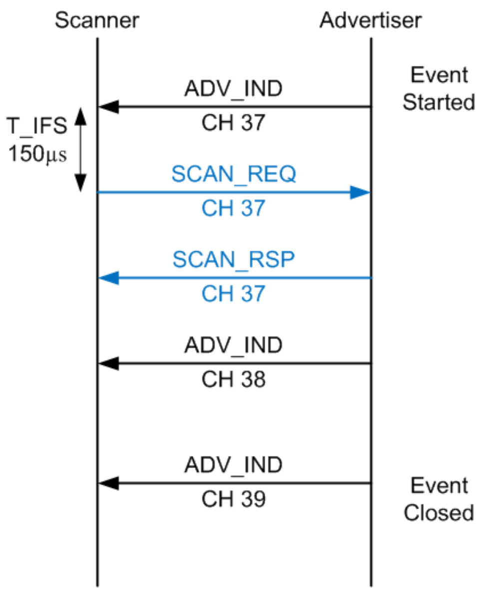

Fig. 2.9: Advertising procedure [19]

2.7.3 Generic Attribute Profile

GATT profile presents another role specification. A server is a device that exposes its data while a Client reads these data. The client is also able to manage activities on Server. GATT Profile specifies data exchange. It is done by specifying the organization of exposed data and ways how to access them. A structure consisting of following building blocks called attributes[4] is presented:

- Service
- Characteristic
- Profile

2.7.4 Profiles

The purpose of the structural architecture of BLE stack might not be visible at first sight, but it ensures Bluetooth's impressive interoperability and robustness. Profiles are protocols that define interactions of layers along the stack. They specify how should two (or more) devices behave with the use of attributes, connections, and safety requests. Bluetooth SIG has implemented various profiles and services that are ready to be used by developers. It contains profiles like Insulin Delivery Profile, Health Thermometer Profile, Heart Rate Profile, etc. Their entire list can be found at [20]. It is also possible to create new custom profiles and services, so the use of SIG adopted ones is not necessary.

2.7.5 Services

Service is a collection of data and associated behaviors to accomplish a function or feature. Contained data are considered as characteristics. Bluetooth SIG has predefined services that developers can use. In the case of WBAN application, service is presented as a Battery Service and WBAN Mode Service.

2.7.6 Characteristics

Characteristic always belongs to certain service and it is the attribute of the lowest level. It contains information elements that the Server exposes to the Client. Elementary information is composed of metadata and actual user data. In terms of example service mentioned above, characteristic would represent a blood sugar value measured by sensor, metadata could be the unit of measurement (mg/ml). This data is exposed to the Client and actions can be made in the opposite direction. One service can include multiple characteristics. Metadata properties and descriptors contain attributes that help with the definition of the characteristic:

- Properties: Read, write, notify, indicate.
- Descriptors: User description, fields used for subscribing to notifications and indications, presentation of the value, etc.

[4] Attribute is piece of data with label

2.7.7 Universally unique identifier

UUID is used to identify services, characteristics, and other attributes, so one device can tell another what service is provided. There are two categories of UUIDs based on its size:

- 16-bit UUID
- 128-bit UUID

16-bit UUIDs are energy efficient in means of data transfer, but they are limited to a relatively small number of unique values due to small data type. That's why are they limited only for Bluetooth SIG attributes. If one decides to create custom services and characteristics, 128-bit UUIDs must be generated and used.

2.8 Proprietary protocols

Proprietary protocols or so-called 3rd party protocols for wireless communication are popular for their simplicity compared to the complex BLE, however, they can only be used in solutions with no smartphones included. Proprietary protocols are owned by a single company and used in its products. This is the main difference from a standard protocol like BLE, accepted by the whole industry. For example, Nordic Semiconductor offers two different proprietary protocols, Gazell and Enhanced-Shock-Burst.

ESB is one of the simplest wireless protocols for basic applications with only the most necessary features and since it does not offer any security management, it will not be further used in this thesis. Gazell shares the same basis with ESB but its more useful since it comes with Gazell Pairing Library that meets the security demands of WBAN device. Security features of this protocol are based on the exchange of secret keys, it splits into a pairing procedure and encrypted data exchange. When two identical WBAN systems are in close distance, miscommunications can happen as a device from one system can send data to the foreign *Host*, therefore pairing procedure is necessary to use.

2.9 Protocol concurrency

Demand for products that support more than one wireless protocol is obvious, as the number of MCU manufacturers offer their portfolio of multi-protocol products. Multi-protocol solutions are promising due to their versatility and readiness for future software refinement with no hardware modification. Various approaches to protocol concurrency are available, choosing the most suitable one depends on the character of application. The main difference is the protocol switching frequency that specifies whether it is a one-time action or periodic event.

2.9.1 Switched multiprotocol

The approach used for situations where two protocols do not have to be online at the same time. If communication protocol needs to be switched, disabling procedures of one protocol are done as well as enabling the procedure of the other one. This process occurs with every protocol exchange. Active communication links are abandoned and have to be recreated every time. The nature of this approach makes it useful for occasional protocol switching in applications like user preparation of IT infrastructure and its resources, this action is called *provisioning*.

An example application can be a configuration of a Wi-Fi thermostat using a smartphone. First, the user connects to the thermostat by Bluetooth communication link and configures its properties (e.g. heating schedules, Wi-Fi network parameters) via front end application. After confirmation of changes, thermostat switches to regular Wi-Fi operation [21].

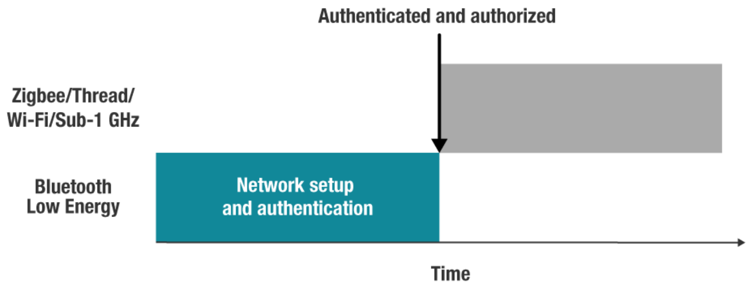

Fig. 2.10: Provisioning example [21]

2.9.2 Concurrent multiprotocol

When the device communicates by two protocols at the same time without breaking the connections, time-division multiplexing of radio time is used. In case when protocols share the same frequency band, one antenna is sufficient. A scheduler is used to handle actions from both protocols based on their priority, avoiding possible jamming yet maintaining created connections. Configurable connection interval allows common transaction events to be performed in the allocated timeslot. This is a suitable approach for application from this thesis, vendor implementation is discussed in chapter 4.4.

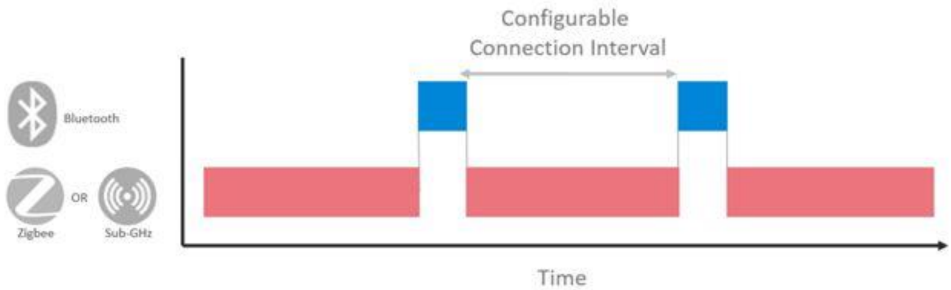

Fig. 2.11: Dynamic multiprotocol [22]

2.9.3 Multi-chip coexistence

When the application demands no compromises in concurrent operation, dedicated hardware for each protocol is implemented. This allows developers to use any combination of protocols with no throughput or frequency limitation. The main disadvantage is higher cost demands as well as a larger PCB footprint.

Chapter 3
Available hardware overview

According to the requirements specified in chapter 2, it is possible to list suitable hardware solutions available in the market. Physical requirements of the system (energy consumption, distance range, device size, etc.) are not specified in numbers, however, these criteria play important role in hardware selection and software optimization since the application is intended to be the battery-operated device in the space-limited environment.

3.1 SiP vs. Chip

Industrial solutions are available as SoCs or as modules, often called System in Package. RF technology is not an easy topic as well as antenna design, so multiple companies make products that integrate SoC with proper on-board antenna design, often accompanied by useful peripherals. One of them can be an external antenna connector, that can be useful in many use-cases utilizing external antenna. SiP modules provide a pre-certified solution with FCC and CE approval, thus ensure rapid integration into commercial application. The use of the smallest SoC packages for SiP modules makes them more reasonable than single-chip solutions even for space-critical requirements.

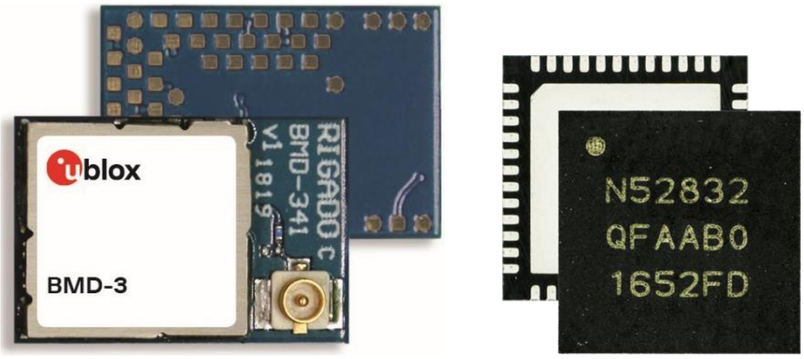

Fig. 3.1: Module and chip hardware

3.2 Vendor product comparison

Nordic Semiconductor specializes in ultra-low power SoC solutions designed for ISM bands like BLE, Bluetooth Mesh, ANT+, Zigbee, etc. During the creation of this thesis, the nRF52 Series family is up to date for BLE applications. SoCs offer hardware needed for Bluetooth 5, feature protocol concurrency with support for 2.4 GHz proprietary stacks. Devices are built on 32-bit *ARM® Cortex™-M4* CPU running at 64 MHz [23]. SiP versions of these chips are available, e.g. from Laird Connectivity.

Another detail worth mentioning is Nordic's well-made customer support consisting of the following web pages that make BLE development more accessible. *Nordic Semiconductor Infocenter* contains online technical documentation of all their solutions and technologies. *DevZone* is Q&A support that also contains blogpost tutorials for getting started.

ON Semiconductor's RSL10 SiP is a direct competitor in the BLE environment with multiprotocol support. It claims the industry's lowest energy consumption in *Deep Sleep Mode* and *Receive Mode* [24]. This product is also attractive because of its very small footprint. STMicroelectronics has a Bluetooth module family called BlueNRG-M with a rather powerful radio [25]. The most important criteria that were used in the hardware selection process are listed in tab.1.

Manufacturer	**Nordic Semi**	**ON Semi**	**STM**
Product	nRF52810 (BMD-3)	RSL10	BlueNRG-M2SA
Flash	192 kB	348 kB	256 kB
RAM	24 kB	72 kB	24 kB
TX Power	+4 dBm	0 dBm	+8 dBm
RX Sensitivity	-96 dBm	-93 dBm	-88 dBm
TX Peak Current	7.0 mA	4.6 mA	10.7 mA
RX Peak Current	4.6 mA	3.0 mA	7.6 mA
DFU OTA[5]	Yes	Yes	No
Antenna	PCB / IPEX MHF4	PCB	Chip
Size [mm]	10.0 x 14.0 x 2.2	8.0 x 6.0 x 1.5	11.5 x 13.5 x 2.3

Tab. 4.1: Hardware Selection Table

[5] This stands for device firmware update over the air. This feature might become handy in case of encapsulated applications with no exposed connectors

BLE product from Nordic Semiconductor is compromising from a technical point of view. nRF52 SoC is used in this thesis, so all the advantages of the mentioned development support can be utilized. This makes the nRF52 based product the ideal candidate for beginner’s BLE embedded development. It is also the most suitable candidate from a financial point of view, which is good for commercial product use.

3.3 Development tools

Bluetooth communication development demands multiple software resources. Tools suitable for Nordic Semiconductor’s portfolio are going to be mentioned in this chapter. A similar software approach can be observed also from other vendors.

Affordable single-board computer with nRF52832 is available for development purposes. It integrates a fair number of GPIO ports with 4 buttons and 4 LED diodes for rapid setup. Arduino *UNO* compliant design makes it possible to use a vast number of shields. J-Link debugger is used for programming and debugging. Power is supplied either by a USB connector or a coin cell battery that makes this development board also a possible demonstrator since it can be detached from a computer.

Another hardware solution offered in a smaller form factor is the nRF *Dongle*. This device similar to an USB stick is a low-cost solution for wireless standards. The dongle can be programmed with a hex file by nRF *Connect for Desktop* application. No debugger is available for this device, which reduces its versatility.

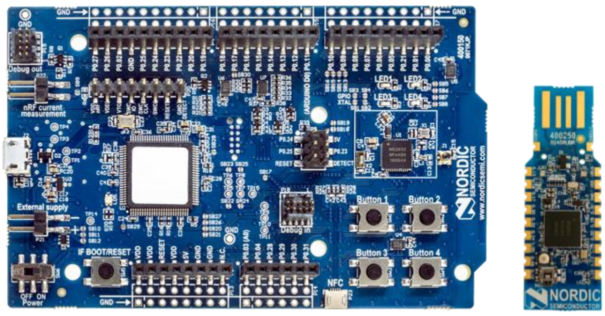

Fig. 3.2: nRF52 development board and nRF *Dongle*

3.3.1 Integrated development environment

- Segger Embedded Studio
- Keil MDK
- IAR Embedded Workbench
- GCC

Four listed IDEs are supported for nRF52 development. Segger Embedded Studio is a license-free environment for ARM Cortex CPUs. It features Real-Time-Terminal debug support, GCC C/C++ compiler, and project management solutions. Segger is also used by Nordic engineers, so there is a chance of future support. These advantages make this IDE suitable for this thesis.

Tools are needed to be able to test applications also from a smartphone point-of-view. Android OS is considered, with version 4.3+ supported. nRF *Connect for Mobile* application or *LightBlue* meet the requirements of this thesis, specifically explore the BLE Services, scan and connect to devices, read and write Characteristics.

3.3.2 Software development kit

SDK is a downloadable zip-archive of software components. It is available for developers to create software that operates on a specific platform. SDK usually consists of:

- Software libraries
- Drivers
- Application Programming Interface
- Integrated Development Environment
- Documentation
- Example applications
- Debuggers, programming tools, etc.

Nordic's SDK contains all these items for setting up peripherals, Bluetooth services, and other functions on nRF52 devices. Header file *sdk_config.h* is part of every project that uses SDK resources. It contains configuration options that manage used modules and libraries. The SDK version used in this thesis is *nRF5_SDK_16.0.0*.

3.4 SoftDevice

Every nRF5x SoC that uses Bluetooth® protocol requires a protocol stack firmware, so it can be used in Bluetooth networks. SoftDevice is a precompiled and linked binary image that provides GAP, GATT, ATT, SM, L2CAP, and Link Layer functionality for BLE application. Therefore, application developers can focus on application instead of Bluetooth configuration at low-level layers. SoftDevice is preprogrammed at an isolated memory location specified by the memory map in fig. 3.3., this location is hardware specific.

SoftDevice API is a set of functions written in C language that is based on Super-Visor-Calls. SVC is a software triggered interrupt handled by SoftDevice IRQ handlers. Calls are used to request resources that are not accessible from the application layer of the program [26]. Chapter 5.4 discusses SVC requests required for the base of this thesis and ways how to access them.

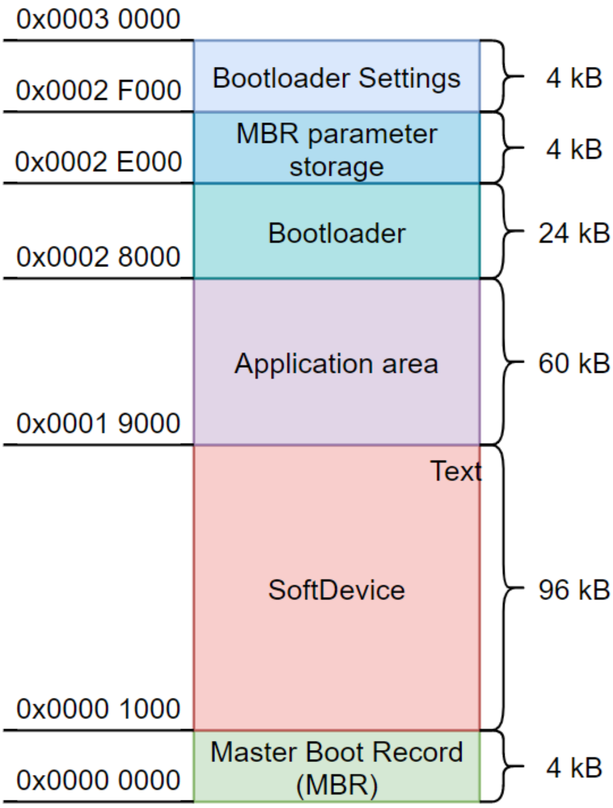

Fig. 3.3: Memory range for nRF52810 with S112 SoftDevice

3.5 J-Link OB

J-link OB allows user to flash and debug programs on Nordic SoC's via Serial Wire Debug interface. This single-chip debug probe is the part of nRF52 development board, OB stands for on-board. Thanks to external Cortex-M Debug Connector, programming custom boards can be simply done without additional programmers. J-link Virtual COM port feature allows communication with computer by USB interface.

Fig. 3.4: Debug probe topology [27]

Chapter 4
Software design

Communication link initialization is introduced in this chapter. The approach focuses on *Device* point-of-view, where most of the features are implemented. Application is evaluated on nRF52 DK that uses nRF52832 chipset. The most recent SDK during the creation of this thesis *nRF5_SDK_16.0.0*, newly offers to use the Gazell protocol on the low-cost nRF52810, as mentioned in its release notes [28]. This hardware is therefore used in custom board design. Due to differences of nRF52 chipsets, nRF52810 will be emulated on development kit to ensure proper functionality after migrating to the target hardware.

The first paragraph of this chapter explains data handling functions. The proprietary protocol interface is made by the Gazell Link Layer. BLE is set up with one custom and one native service. Dynamic multiprotocol feature merges Gazell and BLE so that their concurrent existence is made possible. With all configuration implemented, testing procedures can be specified and performed afterward. Application is merged into SEGGER Solution with the following folder structure:

/wban_device_fw - Root folder contains the following files and folders:

- *main.c* – Main file of the program.
- *wban_ble_gzll_device_pca10040e_s112*.emProject – WBAN *Device* project file
- *sdk_config.h* – Module management
- *flash_placement.xml* – Flash memory addresses

/data - Data handling functions presented according to specification in chapter 1.

- *battery_level.h* – Declarations for battery measurement
- *battery_level.c* – ADC setup and battery data processing
- *modes_wban.h* – Declarations for mode select
- *modes_wban.c* – Mode structure components

/services - Initialization of services according to the BLE stack requirements.

- *mode_service.h* – Declarations for Mode Select Service
- *mode_service.c* – Custom Mode Select Service configuration

/multiprotocol - Timeslot API configuration with Gazell startup procedure

- *timeslot_gzll.h* – Declarations for Timeslot API
- *timeslot_gzll.c* – Timeslot API configuration with Gazell startup

/backup – Custom board headers backup

- *wban_device_brd.h*
- *wban_host_brd.h*

4.1 Data handling

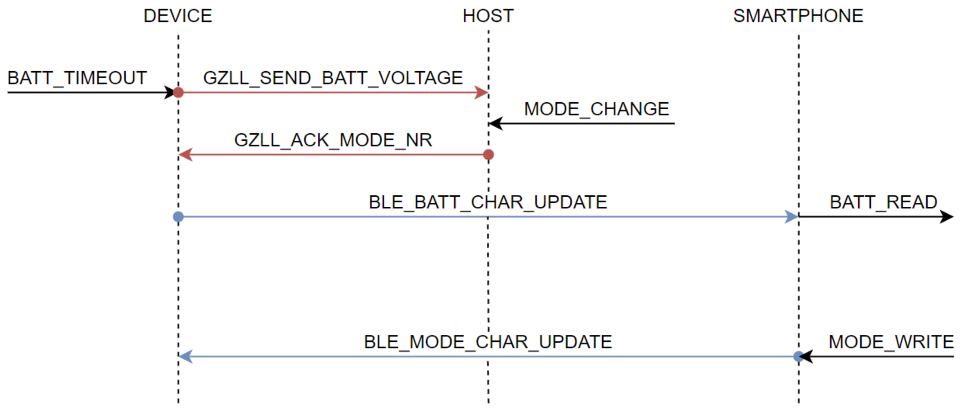

Fig. 4.1: Data exchange flowchart

Fig. 4.1. presents the way data flows in the application. Data exchange through the Gazell link is synchronized by repeated timer instance set up in the *Device*. When the timer expires, the voltage at certain analog input is measured. When the battery reading is ready, *Device* sends data to the *Host* by Gazell link. When the *Host* receives the data, it acknowledges successful transfer with the packet containing *WBAN mode* number selected by the user. *WBAN mode* can be changed on the *Host* side anytime during the operation, but the data is always sent with the next acknowledgment packet. Update of Battery Service Characteristic value is also part of the battery timer interrupt event, in case of existing BLE connection. The *WBAN mode* number change and battery data access actions from the smartphone can be performed anytime by writing to BLE Characteristics independently of battery timer interrupts.

4.1.1 App Timer instances

Application Timer API is an easy way of implementing timer instances based on RTC peripheral with a low-frequency-clock source. Both repeated and single-shot timers are supported. Instances are created for battery measurement, battery voltage indication, and button debouncing. Initialization of timer instance begins with allocating required memory and creating its identifier by calling:

```
APP_TIMER_DEF(m_battery_timer_id);                    /**< Battery timer instance. */
```

This ID is referenced in each function related to the certain timer instance. Timer module is initialized and after that, the referenced timer instance can be created. Timeout handler and mode type are requested inputs. After successful initialization, the referenced timer can be started with a certain interval between timeouts as an input. This initialization approach is similar to the remaining App Timer instances used in the WBAN application.

```
// Initialize timer module.
app_timer_init();

// Create battery measure timer.
app_timer_create(&m_battery_timer_id,
                 APP_TIMER_MODE_REPEATED,
                 battery_meas_handler);

// Start timer
app_timer_start(m_battery_timer_id, BATTERY_LEVEL_MEAS_INTERVAL, NULL);
```

4.1.2 Motor driver

Vibration motor used for haptic feedback is driven by a PWM signal that allows a customizable vibration profile for different types of events. Nordics SoC contains dedicated PWM hardware so there is no need for building the modulated signal with other peripherals (TIMER, PPI, GPIOTE). Low-cost version nRF52810 implements just one PWM generator. This means that only one instance can be created. The instance can control 4 output channels at the same time with one PWM signal frequency. WBAN application does not require more than one instance at one moment and the frequency of the PWM signal can be reconfigured during the *WBAN mode* switching.

```
nrf_drv_pwm_config_t const config1 =
{
    .output_pins =
    {
        BSP_LED_0 | NRF_DRV_PWM_PIN_INVERTED, // channel 0
        BSP_LED_1 | NRF_DRV_PWM_PIN_INVERTED, // channel 1
        MOTOR_PIN                           , // channel 2
        NRF_DRV_PWM_PIN_NOT_USED              // channel 3
    },
    .irq_priority = APP_IRQ_PRIORITY_LOWEST,
    .base_clock   = NRF_PWM_CLK_125kHz,
    .count_mode   = NRF_PWM_MODE_UP,
    .top_value    = COUNTER_TOP,
    .load_mode    = NRF_PWM_LOAD_INDIVIDUAL,
    .step_mode    = NRF_PWM_STEP_AUTO
};
```

PWM peripheral has dedicated driver *nrfx_pwm.c* that can be used for setup matching application needs. Initialization begins with mapping output pins with PWM channels. Here, inverted polarity can be defined as for the LEDs in the code snippet above. The increment is either middle or edge centered. COUTER_TOP value together with PWM clock defines signal period according to equation (4.1), considering edge centered counter.

$$T_{PWM} = \frac{COUNTER\_TOP}{f_{PWM}} \tag{4.1}$$

The duty cycle is configured by updating the compare register. PWM driver updates this register by calling the function *nrfx_simple_playback*. Values for compare registers are stored as a sequence in array *seq1_values*. Duty cycles are fractions of COUNTER_TOP values. Four types of accessing the sequences are enumerated below. Individual mode is suitable for WBAN application since each channel can have its own sequence.

```
/**
 * @brief PWM decoder load modes.
*/
typedef enum
{
    NRF_PWM_LOAD_COMMON
    NRF_PWM_LOAD_GROUPED
    NRF_PWM_LOAD_INDIVIDUAL
    NRF_PWM_LOAD_WAVE_FORM
} nrf_pwm_dec_load_t;
```

Vibration motor feedback is called in the case of U_{BAT} < 3000 mV considering discharging events. The sequence of two short pulses with 320ms period is played twice, following an 800ms break. PWM period is set to 80ms by COUNTER_TOP equal to 10000 and PWM clock pre-scaled to f_{PWM} = 125 kHz.

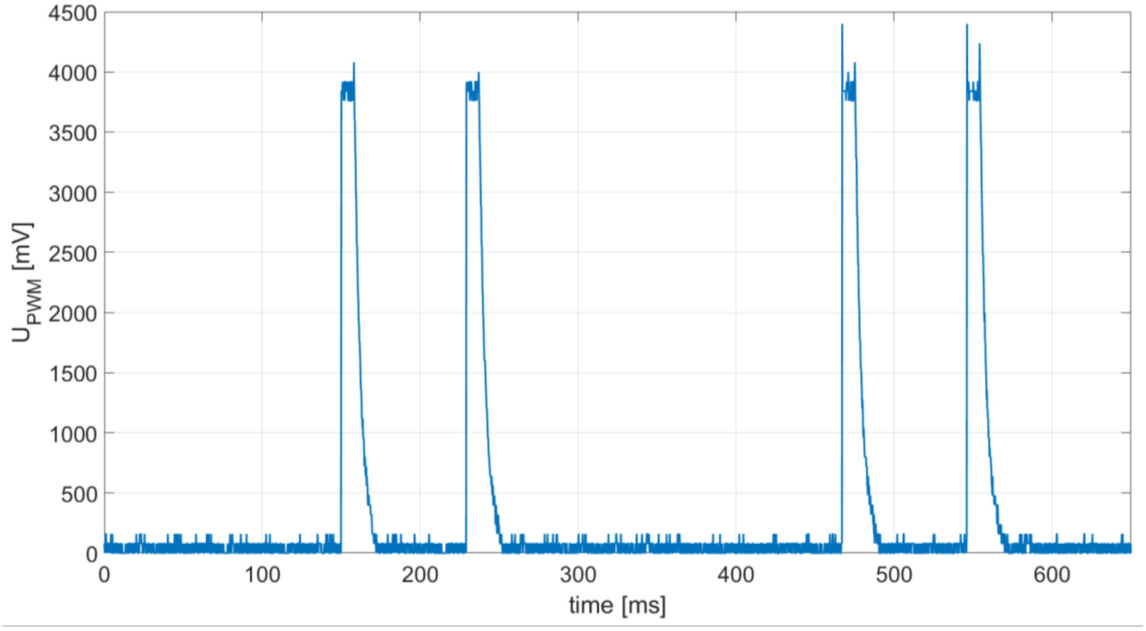

Fig. 4.2: PWM signal at motor terminals

4.1.3 SAADC module

To obtain the voltage at analog input of the SoC, SAADC is used. The first thing to be defined is a reference voltage with two choices available. First is the fixed internal reference, second is the V_{DD} reference relative to the supply voltage. Obtained signals from the sensor nodes need to be independent of fluctuating input voltage in case it is battery operated. Therefore, the internal reference is selected. Internal reference sets the input range of the ADC core to U_{REF} = ±0.6V. However, this is not equal to the range at the input to the SoC. This interval called dynamic range must not exceed $\pm V_{DD}$, which is done by using a voltage divider (discussed in chapter Custom Board Design). The gain setting is the parameter that scales the voltage from the dynamic range into the range of the ADC core. 10-bit ADC with maximum input voltage U_{ADC} = 3V at input pin can resolve measurement in the range of 1024, with a resolution of U_{RES} = 2,92 mV. Macro ADC_RESULT_MV() converts the digital value from SAADC into millivolts by using equation (4.2). The voltage reading is stored in 16-bit unsigned integer *vbatt*.

$$ADC\_RESULT\_MV = \frac{ADC\_VALUE * ADC\_REF\_MV}{ADC\_RES\_10BIT} * ADC\_GAIN \tag{4.2}$$

ADC conversion is driven by App Timer API working in repeated mode. Timeout handler *battery_meas_handler* is called every 1200 ms. In the handler function, *battery_meas_get* function is called to prepare SAADC conversion and measure the single-ended signal at the AIN2 pin. Voltage is processed by SAADC. As soon as the conversion is over, *saadc_event_handler* is called, and stored data can be accessed.

```
/**@brief Function handling events from 'nrf_drv_saadc.c'.
 *
 * @param[in] p_evt SAADC event.
 */
static void saadc_event_handler(nrf_drv_saadc_evt_t const * p_evt)
{
    if (p_evt->type == NRF_DRV_SAADC_EVT_DONE)
    {
        nrf_saadc_value_t adc_result;

        adc_result = p_evt->data.done.p_buffer[0];

        m_batt_lvl_mv = ADC_RESULT_MV(adc_result);
    }
}
```

Measured voltage data have to be processed into the format requested from each communication channel. Gazell protocol sends 1-byte payloads only, so 16-bit integer *vbatt* is split into two 8-bit parts stored in field *gzll_packet*.

```
// Split 16-bit vbatt into 8-bit packets for gzll transfer
gzll_packet[0] = (uint8_t)((vbatt & 0xFF00) >> 8);
gzll_packet[1] = (uint8_t)(vbatt & 0x00FF);
```

Native Bluetooth SIG Battery Service has a mandatory format of data. It is the battery level as a percentage from 0% to 100%, 0% represents a battery that is fully discharged, 100% represents a battery that is fully charged. That is why the *vbatt* is converted into the state of charge *vbatt_percent* by the discharge curve discussed in [29]. Variable *vbatt* serves as an input to the function *battery_percent_lipol*. The structure is made from data of the Li-pol battery discharge curve 1C discharge, temperature difference over time is not considered.

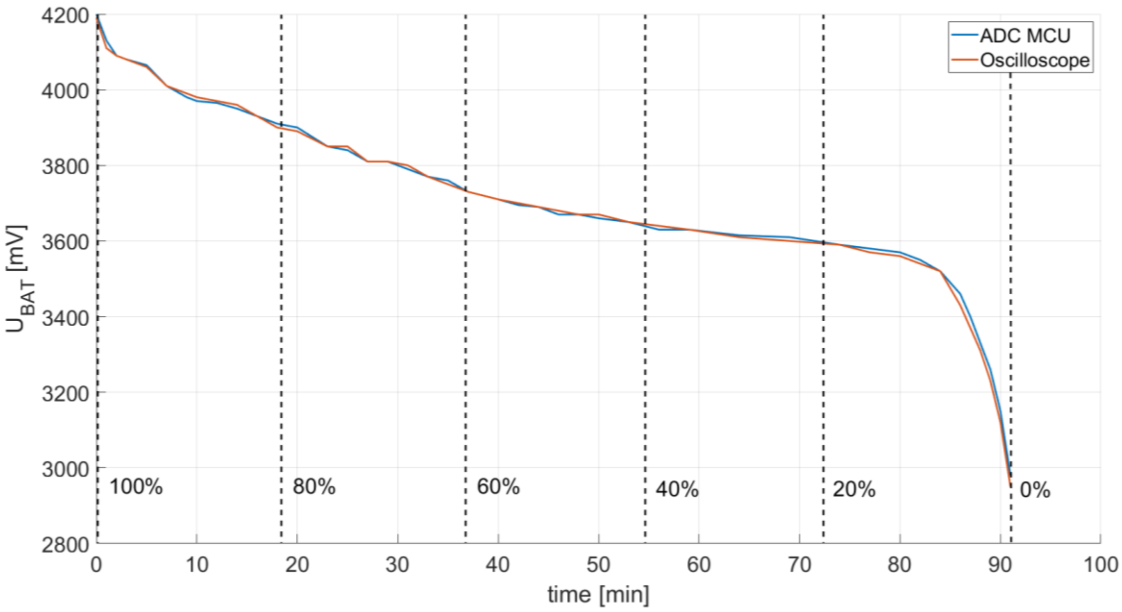

Fig. 4.3: Li-pol battery discharge curve [29]

4.1.4 WBAN mode select

This library contains functions for operation modes management. This includes signalization by on-board peripherals and switching events by buttons. Modes are defined as enumeration type *modes_wban_t*. Indexing of modes with numbers is used in wireless data exchange, *actual_mode* variable stores the index value that is the currently used mode and content of transmitted data packet.

```
/**@brief Mode select states.
*/
typedef enum
{
    MODE_FIRST = 0,
    MODE_READY = MODE_FIRST,
    MODE_ON,
    MODE_PWM,
    MODE_MOT,
} modes_wban_t;
```

Indexed modes have assigned certain functionality defined in *mode_indication* function. Tab. 4.1 contains the peripheral actions based on the selected mode. Unique signalization is helpful for system testing and debugging purposes. *WBAN Mode* MODE_READY is set by default after the startup of *Device*, it displays the BLE and multiprotocol functionality in case of successful initialization.

Mode name	**MODE_READY**	**MODE_ON**	**MODE_PWM**	**MODE_MOT**
Mode nr.	0	1	2	3
LED1	BLE	ON	PULSE	OFF
LED2	Multiprotocol	ON	OFF	ON
MOTOR	OFF	OFF	PULSE	ON

Tab. 4.1: Mode signalization

4.1.5 Button configuration

The WBAN mode number can be written as characteristic value via smartphone, this topic is further discussed in chapter *WBAN Mode* Service. *WBAN Mode* number can be also changed by button press from *Host*. Function *button_event_handler* function is called every time button press action is detected. Both ascending and descending change in index value shall be possible, so the *wban_mode_t* can be cycled through in both directions. Two tactile buttons are required to allow such a feature.

```
switch (pin_no)
{
    case BUTTON_UP:
        if (button_action == APP_BUTTON_PUSH)
        {
            if (actual_mode != MODE_FIRST)
                actual_mode--;
            else
                actual_mode = MODE_MOT;
        }
        break;
    case BUTTON_DN:
        if (button_action == APP_BUTTON_PUSH)
        {
            if (actual_mode != MODE_MOT)
                actual_mode++;
            else
                actual_mode = MODE_FIRST;
        }
        break;
    default:
        return;
}
```

Button pin defined as BUTTON_*xx* is the input to the function that decides if *actual_mode* variable should be increased or decreased. If the end of the list is reached, it jumps back into the beginning and vice versa. APP_BUTTON_PUSH action is falling edge detection, other actions (e.g. long push, pull) are other possibilities to use.

Reliable operation of button actions is possible with debouncing. A single-shot timer instance is started when the push action from the button is detected. After the timer expires, the state of the pin is checked again whether it is still in the active state. If so, the button press is reported by calling the handler *button_event_handler*.

4.2 Gazell Link configuration

Role assignment is the first step in the Gazell protocol configuration. Two roles are offered, *Host* and *Device*, while the *Device* is always being an initiator of communication. *Host* acknowledges received packet and it is also possible to piggyback 32 bit of user data into the acknowledgment packet. Gazell offers a star network where 8 devices can communicate with a single *Host*. This protocol is designed for applications where one side of communication does not have strict power constraints. It is because the *Host* is always listening for an incoming packet, to ensure rapid feedback. *Device* mode is more energy efficient; it will be assigned onto a body part [30]. Roles are set in the application as the input to the *nrf_gzll_init* function. Transmitter power is set to the peak value, +4dBm. After successful initialization is complete, data transfer can be enabled.

```
/**@brief Initialization procedure of Gazell Link
 */
nrf_gzll_init(NRF_GZLL_MODE_DEVICE);

nrf_gzll_set_device_channel_selection_policy(
                    NRF_GZLL_DEVICE_CHANNEL_SELECTION_POLICY_USE_CURRENT);

nrf_gzll_set_xosc_ctl(NRF_GZLL_XOSC_CTL_MANUAL);

nrf_gzll_set_max_tx_attempts(100);

nrf_gzll_set_base_address_0(0xE7E7E7E7);

nrf_gzll_set_tx_power(NRF_GZLL_TX_POWER_4_DBM);

nrf_gzll_enable();
```

Whenever the data is ready to be transmitted from the *Device* side, function *nrf_gzll_add_packet_to_tx_fifo* is called. This function requires the channel used for unencrypted data transfer as an input. This is a table of frequencies called pipe used to avoid interference in case more devices communicate with one *Host.* PIPE_NUMBER is set to 0. If the *Host* receives the data packet successfully, callback function *nrf_gzll_host_rx_data_ready* returns TRUE value so data can be accessed and acknowledged. The frequency of data exchange is driven by App Timer interrupts.

4.3 BLE Services

Two services are mandatory for all Bluetooth devices: GAP and GATT Service. GAP service contains characteristics like device name, appearance, connection parameters, etc. These parameters are configured in *gap_params_init* function. GATT service is used to notify that attribute structure changed on the peripheral device. Neither of these two services is included in the advertising packet.

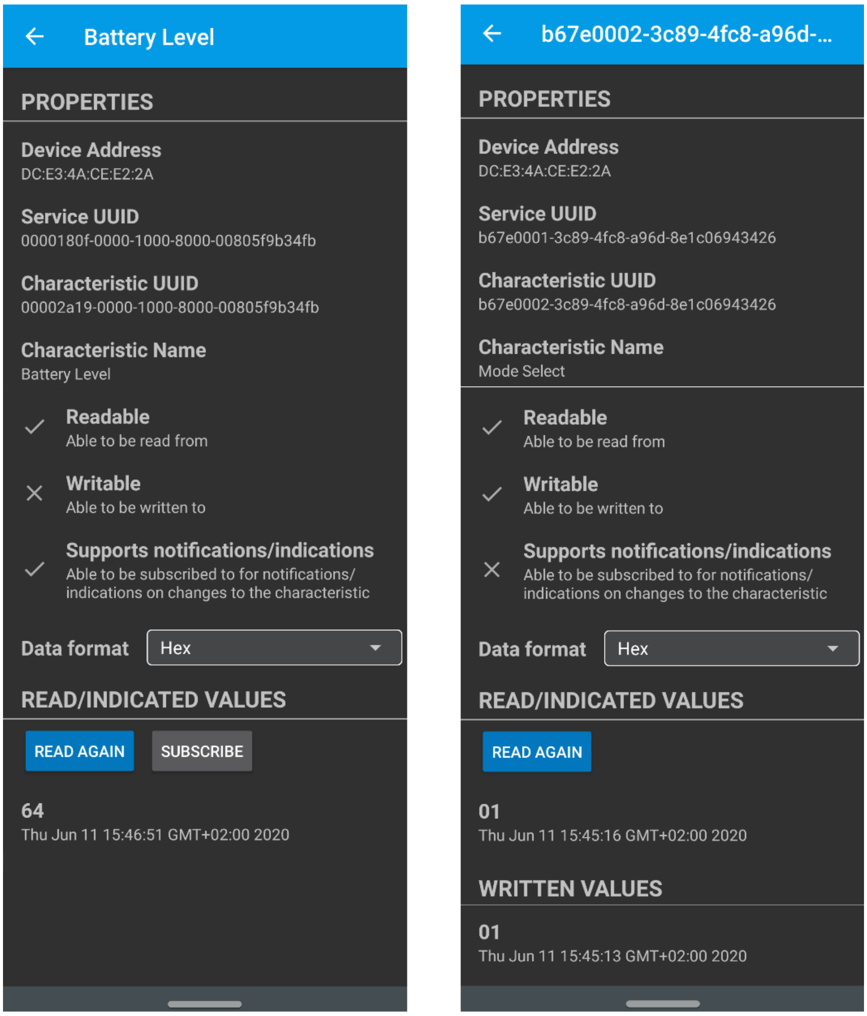

Fig. 4.4: Services observed from the smartphone

Since none of the native Bluetooth services fit the requested functionality, custom service is created upon *WBAN Mode* selector. First, a 128-bit integer number is generated using a tool from [31]. To create base UUID, bytes highlighted in code snippet are replaced with zeros. These bytes are reserved for Service and Characteristic UUIDs, that will share the same base. The rule in base UUID generating is not to collide with base UUID reserved for native services. The following steps display the process of UUID creation.

Custom service initialization contains adding the custom base UUID to the SoftDevice's internal list by calling *sd_ble_uuid_vs_add*. Function outputs the pointer *p_uuid_type* to an 8-bit unsigned integer, where the type field in *ble_uuid_t* corresponding to this UUID will be stored. This returned value has to be assigned to every attribute that uses the specific base UUID. Custom service is added to the attribute table by calling *sd_ble_gatts_service_add*.

```
B67E4E5C-3C89-4FC8-A96D-8E1C06943426     // Generated number
B67E0000-3C89-4FC8-A96D-8E1C06943426     // Base UUID
B67E0001-3C89-4FC8-A96D-8E1C06943426     // WBAN Mode Service UUID
B67E0002-3C89-4FC8-A96D-8E1C06943426     // WBAN Mode Characteristic UUID
```

WBAN Mode Service contains one characteristic that will be used to hold the index of the selected *WBAN mode*. Function *mode_nr_char_add* is called to initialize properties of this characteristic. Both read and write permissions are configured together with another UUID procedure like the one for *WBAN Mode* Service. *Char_md.p_char_user_desc* holds the name of this characteristic, that is visible if the characteristic is observed from the server. Attribute value settings contain information about the size of the transmitted user data packet. In this case, 1 byte of data is selected. *WBAN Mode Select* Characteristic is added to the attribute table by calling *sd_ble_gatts_characteristic_add* [34].

```
/**@brief Function for adding the WBAN Mode characteristic.
*/
static uint32_t mode_char_add(ble_mode_service_t * p_mode_service)
{
ble_gatts_char_md_t char_md;
ble_gatts_attr_t    attr_char_value;
ble_gatts_attr_md_t attr_md;
ble_uuid_t          ble_uuid;

// Define the WBAN Mode Characteristic properties and name
char_md.char_props.read          = 1;
char_md.char_props.write         = 1;
char_md.p_char_user_desc         = ModeCharName;

// Define the WBAN Mode Characteristic UUID
ble_uuid.type = p_mode_service->uuid_type;
ble_uuid.uuid = BLE_UUID_MODE_CHAR_UUID;

// Set permissions on the Characteristic value
BLE_GAP_CONN_SEC_MODE_SET_OPEN(&attr_md.write_perm);
BLE_GAP_CONN_SEC_MODE_SET_OPEN(&attr_md.read_perm);

// Attribute Value settings
attr_char_value.p_uuid       = &ble_uuid;
attr_char_value.init_len     = sizeof(uint8_t);
attr_char_value.max_len      = sizeof(uint8_t);
```

The function that handles write events from the Server to this characteristic assigns the correct operation mode based on the received mode number. *mode_change_handler* function is called every time the BLE_GATTS_EVT_WRITE event will occur. The function is a case structure that contains assigned functionality for peripherals (2xLED, vibration motor) in the application [34].

Battery Service uses code provided by the vendor in *ble_bas.c*, so implementation is as simple as using initialization and update functions. When the voltage measure timeout occurs, the battery level characteristic is updated by calling *ble_bas_battery_level_update*, in case the server is connected. The connection condition avoids useless data transmission without the receiver. The data format for battery Service must be the state of charge in percent, this is already discussed in chapter 4.1.3.

4.3.1 BLE advertising

Advertising structure is configured to create a connectable undirected device of BLE only type. This means that any device supporting Bluetooth 4.1 can send scan request or connect request when advertising is on. *Device* broadcasts data called advertising packet during advertising event in specified interval and duration, these are the parameters configured in *init.config* structure.

Fig. 4.5: Advertising data observed from the smartphone

When *Device* receives the scan request, scan response packet is sent. Both mentioned packets can contain 31 bytes of data. Following list contains information that is advertised with its size in bytes:

- Flags (LE only) 3
- Appearance 1
- Name (WBAN_node) 9
- Battery service UUID 4
- Mode Select service UUID 18

The sum of the information bits size is 35 bytes. This means it must be split into advertising and scan response packet since one of them cannot fit all. Following code presents the complete advertising initialization:

```
/**@brief Function for initializing the Advertising functionality.
 */
static void advertising_init(void)
{
    ble_advertising_init_t init;

    init.advdata.name_type          = BLE_ADVDATA_FULL_NAME;
    init.advdata.include_appearance = true;

    init.advdata.flags = BLE_GAP_ADV_FLAGS_LE_ONLY_GENERAL_DISC_MODE;

    init.srdata.uuids_complete.uuid_cnt = sizeof(m_adv_uuids) /
                                          sizeof(m_adv_uuids[0]);
    init.srdata.uuids_complete.p_uuids  = m_adv_uuids;

    init.config.ble_adv_fast_enabled  = true;
    init.config.ble_adv_fast_interval = APP_ADV_INTERVAL;
    init.config.ble_adv_fast_timeout  = APP_ADV_DURATION;

    init.evt_handler = on_adv_evt;

    ble_advertising_init(&m_advertising, &init);

    ble_advertising_conn_cfg_tag_set(&m_advertising, APP_BLE_CONN_CFG_TAG);
}
```

4.4 Timeslot API

Concurrent operation of BLE and Gazell is set up by Timeslot API. *Timeslot* is a period where access to radio peripherals is given to application needs, in this case, it is another wireless communication protocol. To be able to handle timeslots, SoftDevice API function has to be called. After that, timeslot can be requested within a session.

```
sd_radio_session_open(radio_callback);
sd_radio_request(&m_timeslot_request);
```

Communication between Timeslot feature and the application is done by events and signals. Events are used for session management. These come from SoftDevice and give status about the current session, *nrf_evt_signal_handler* is the function that manages events inside the WBAN application. By default, the application is not able to see the events, so their handling fails. This is solved by registering the event handler as an Observer by calling a following macro:

```
NRF_SDH_SOC_OBSERVER(m_time_slot_soc_observer, 0, timeslot_evt_handler, NULL);
```

NRF_EVT_RADIO_CANCELED status event is called when timeslot was canceled because of overlap with higher priority events. The application has to request a new timeslot in this situation. Other events have no further implementation needed.

Timeslot signals are from radio peripherals. Signals give status about the current timeslot state and WBAN application manages them in *radio_callback* function. At the beginning of each timeslot, NRF_RADIO_CALLBACK_SIGNAL_TYPE_START callback is received. This is where TIMER0 is started to trigger event before timeslot ending, ensuring that operations inside timeslot manage to shut down properly. Excluding this procedure can result in unexpected behavior from SoftDevice. Gazell is initialized every time in the timeslot entrance, this is done in the *m_on_start* function. After initialization, Gazell works as described in chapter 4.2 during the timeslot window.

```
/**@brief Function called before ending of the timeslot
*/
static void m_on_multitimer(void)
{
    NRF_TIMER0->EVENTS_COMPARE[0] = 0;
    if (nrf_gzll_get_mode() != NRF_GZLL_MODE_SUSPEND)
    {
        signal_callback_return_param.params.request.p_next = NULL;
        signal_callback_return_param.callback_action =
                                        NRF_RADIO_SIGNAL_CALLBACK_ACTION_NONE;
        (void)nrf_gzll_set_mode(NRF_GZLL_MODE_SUSPEND);
        NRF_TIMER0->INTENSET = TIMER_INTENSET_COMPARE0_Msk;
        NRF_TIMER0->CC[0] = m_slot_length - 1000;
    }
    else
    {
        ASSERT(nrf_gzll_get_mode() == NRF_GZLL_MODE_SUSPEND);
        configure_next_event_normal();
        signal_callback_return_param.params.request.p_next = &m_timeslot_request;
        signal_callback_return_param.callback_action =
                          NRF_RADIO_SIGNAL_CALLBACK_ACTION_REQUEST_AND_END;
    }
}
```

NRF_RADIO_CALLBACK_SIGNAL_TYPE_TIMER0 is the signal that is handled before the timeslot ending by *m_on_multitimer* function. Here, shutdown procedures are started, and the next timeslot is requested. Gazell disabling can be easily done by setting the mode to NRF_GZLL_MODE_SUSPEND.

Chapter 5
Custom hardware design

The hardware design process is summarized in this chapter, starting with the description of each subsystem. Battery management for selected power supply is designed. Custom PCB integrates the WBAN communication interface with wireless power transfer solution from [29] so that the whole system can be demonstrated as one.

5.1 Power supply

WBAN Device as well as *Host* are equipped with rechargeable batteries. Li-pol chemistry is selected due to its large energy density and durability. On top of that, this type of battery is available in prismatic shape suitable for space-critical applications. Multiple choices of battery capacity are offered so that the most suitable one can be found for each application. Choosing the right battery parameters is based on the following criteria:

- Minimal battery life
- Size
- Recharge time

Battery capacity is proportional to battery life in case of the constant current load. The energy consumption of the *Device* is measured in chapter 6.5. Results from the current measurement are often simplified by the substitution for a constant current value. However, large current spikes caused by alternating sleep mode periods make this simplification incorrect. These spikes are common for wireless communication solutions used in IoT. Battery loaded with constant current operates longer than battery loaded with current spikes [32]. Precise battery capacity optimization would require simulation of various consumption profiles with no guaranteed result. The simplified approach used in this thesis is presented. Required battery life is set to a minimum of t_{BAT} = 16h in chapter 1 requirements. Minimal capacity is calculated using constant current load substitution equation (5.1).

$$Q_{BAT} = t_{BAT} \cdot i_{CC} \tag{5.1}$$

Calculated capacity Q_{BAT} defines the lower limit of the suitable batteries range, ones with the capacity greater than Q_{BAT} can be used. The upper limit is set according to the mechanical size criteria if there are any. The battery is chosen by recharge time criteria because charging current is limited to 200 mA and the battery has to be charged within 2 hours. Using this selection process, a 200 mAh battery is selected.

5.2 Battery management

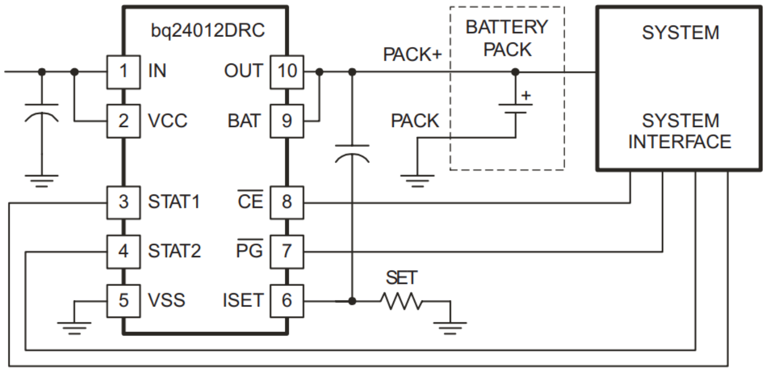

Fig. 5.1: Battery charger schematics

Li-pol battery chargers operate in CC/CV mode. This means that charging consists of three phases, conditioning, constant current, and constant voltage [33]. Fig. 5.1 presents charge management IC *bqTINY*^{TM} in the *QFN* package. It features current and voltage regulation, status indication, and short-circuit protection. Version bq24012 is used due to its charge enable feature, controlled by $\overline{\text{CE}}$ input. The control interface is standalone, the regulated output voltage is $V_{O(REG)}$ = 4.2 V by default. Charging current limit $I_{O(OUT)}$ = 200 mA is set by R_{SET} resistor value according to (3)

$$R_{SET} = \frac{K_{SET} \cdot V_{SET}}{I_{O(OUT)}} \tag{5.2}$$

Status pins give information about the charging process and valid input power presence. They will be used at the PCB prototype for LED signalization. STAT1 pin is in ON state in case of active battery charging. STAT2 pin is ON when the charging procedure is finished successfully. In the case of both STAT pins in OFF state, situations listed below can happen. The actual situation can be discovered by monitoring states at pins IN, BAT, $\overline{\text{CE}}$, and $\overline{\text{PG}}$.

- Absence of battery
- Temperature protection
- Sleep mode
- Damaged battery

5.3 nRF52810

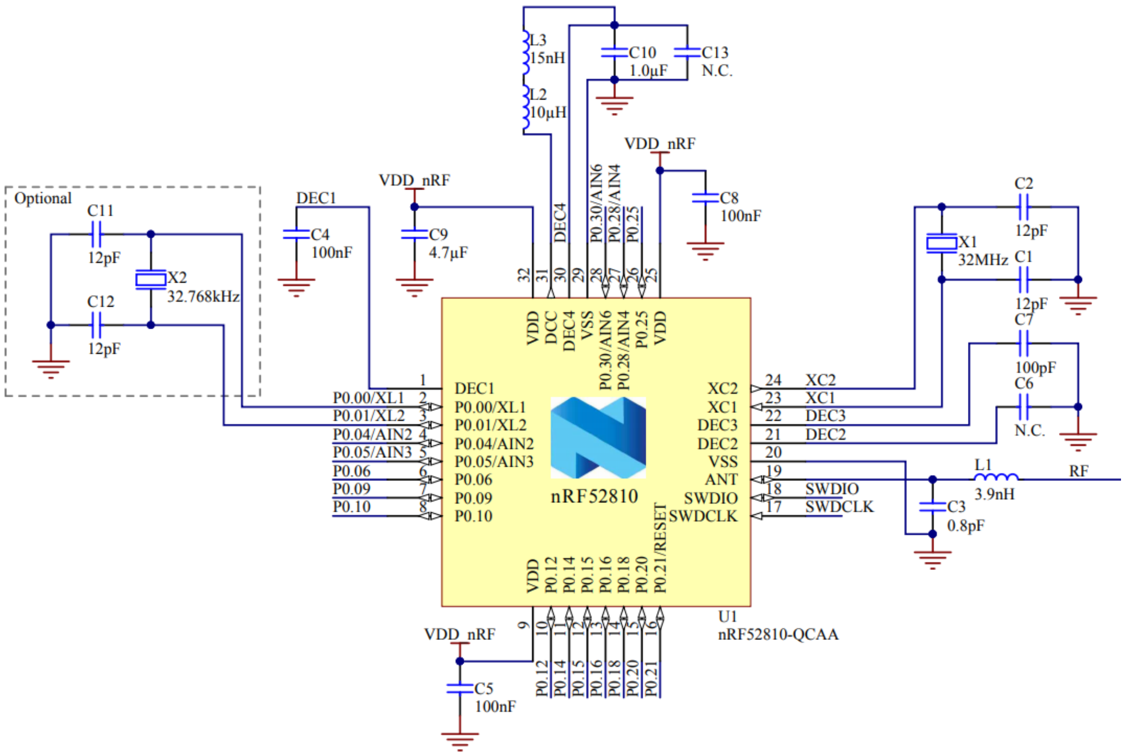

Fig. 5.2: nRF52810 reference layout

Fig. 5.2 presents a reference layout of 2.4 GHz transceiver in a *QFN32* package. This layout contains two external clock sources, crystal X1 with frequency f_{X1} = 32 MHz and X2 with frequency f_{X2} = 32,768 kHz. Matching load capacitors C1, C2, and C11, C12 ensure accurate oscillation frequency of X1 and X2. LC filter connected to the DEC4 pin (L2, L3, C10) allows the user to use an internal DC/DC regulator instead of a default internal LDO regulator. This feature is an optional part of custom PCB design and has to be enabled in software. DC/DC regulator is for testing purposes only due to the planned usage of external power management solution. LC circuit created from components C3 and L1 is a matching circuitry for the antenna with impedance Z_{ANT} = 50 Ω. Capacitors connected to DEC pins decouple power supply, as well as input capacitors C5, C8, C9. The recommended supply voltage V_{DD} = 3V is connected to three VDD input pins. SWDIO and SWDCLK pins provide an SWD interface for device programming. SWO pin that serves as debug output is only present in *QFN48* versions of this SoC, but it is not necessary for the device programming. The rest of the pins marked with P0 prefix can be used as general-purpose inputs/outputs. Selected GPIO pins with AIN suffix can be configured as analog inputs.

5.4 Motor driver

The motor is powered directly from the battery and not from the external 3V3 LDO. This is because LDO with low voltage drop and limited output current is used. Motor current draw could cause malfunctions or overheat of the mentioned part. PWM signal from SoC serves as an input to the gate of *N-Channel MOSFET* that switches the motor in low-side transistor configuration. Due to the fast switching signal, the Schottky diode is connected in parallel to the motor terminals. This provides a safe path for inductive kicks from the motor switch off due to its inductance, protecting the switching transistor. A small ceramic capacitor connected in parallel to the motor reduces RF noise caused by current spikes.

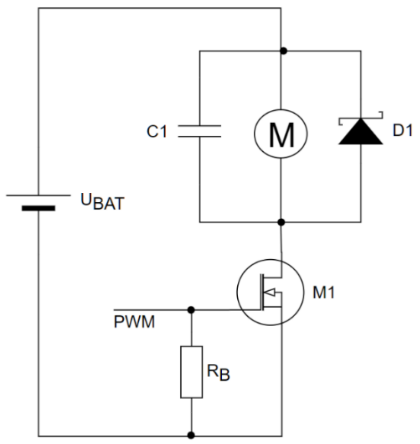

Fig. 5.3: Haptic motor hardware driver schematics

5.5 Signal measurement

A single-ended signal that exceeds limits of SoC pin input requires a simple hardware setup to prevent chip damage. Battery level is the measured signal in this case, a similar approach can be used on any other sensor node. The peak voltage that can be exposed to GPIO pin is $V_{MAX} = (V_{DD} + 0.3)$ V, larger voltage signals are divided to fit recommended operating conditions. Voltage divider with the capacitor connected in parallel is used as shown in fig. 5.4. To prevent significant current leak through the voltage divider, high-value resistors are chosen.

SAADC used in nRF52 devices samples input signal by connecting capacitive load (Sample and hold capacitance) to the analog input pin for a specified amount of time. The required acquisition time of measured signal t_{ACQ} depends on input impedance. In case divider impedance is too high, acquisition time has to be extended. High impedance at the input is also more sensitive to noise. To solve these issues, external capacitor CEXT is added in parallel with the voltage divider. CEXT is charged to voltage level U_{DIV} according to the voltage divider properties. During the acquisition, the charge is spread between C_{EXT} and C_{SAMPLE} in minimum time, so t_{ACQ} does not have to be extended.

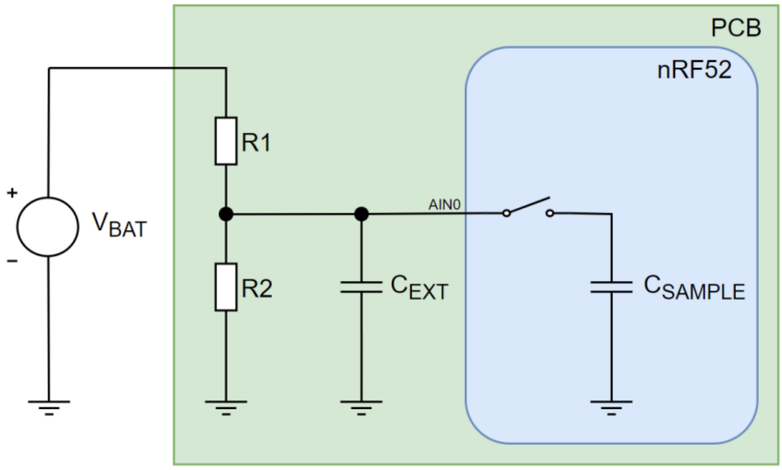

Fig. 5.4: Analog input hardware protection

Maximum voltage on Li-pol battery cell, V_{BAT} = 4.2 V, exceeds analog pin input limit, this voltage is scaled to U_{DIV} = 3 V. External divider impedance must be much lower than Z_{IN} value given by eq. 5.3, where f_{SAMPLE} is the sampling frequency and C_{SAMPLE} = 2.5 pF. In case of f_{SAMPLE} = 10 Hz, input impedance Z_{IN} = 40 GΩ.

$$Z_{IN} = \frac{1}{f_{SAMPLE} \cdot C_{SAMPLE}} \tag{5.3}$$

The upper resistor is selected to be R_1 = 800 kΩ. The R2 value is calculated by (5.4) to be R_2 = 2 MΩ. External capacitor value is set to C_{EXT} = 2.5nF. These values are low enough to ensure the correct operation of SAADC yet provide low current losses.

$$R_2 = \frac{R_1 \cdot U_{BAT}}{U_{BAT} - U_{DIV}} \tag{5.4}$$

5.6 Pin assignment

A custom board pin assignment is specified to enable the functionality of peripherals. The header file that includes all definitions of peripheral pins used on the custom board is created. The idea behind this is to eliminate code changes when migrating to different hardware (e.g. development board). The only action to be done is changing the current board file in options. Both *Host* and *Device* PCBs have their own header file.

- *wban_device_pcb.h*
- *wban_host_pcb.h*

All board files are stored inside SDK's components folder by default, custom headers are added here as well. Boards are also included in *boards.h* file so that they can be accessed through the options tab.

```
/** @brief File includes inside boards.h
 */
#elif defined (BOARD_WBAN_DEVICE_PCB)
  #include STRINGIFY(wban_device_pcb.h)
#elif defined (BOARD_WBAN_HOST_PCB)
  #include STRINGIFY(wban_host_pcb.h)
```

Modifications made outside the project folder create problems with project migration to another SDK (e.g. different PC). A good manner is to include backup board files in the project folder in case of modifying this WBAN project at different SDKs. Peripheral assignment defines are written by certain rules so that other modules can interact with the hardware properly. LEDs, buttons, motor, and ADC measurement pins are defined inside custom files. The following code snippet is taken from *wban_device_pcb.h* file.

```
// LEDs definitions
#define LEDS_NUMBER         2
#define LEDS_ACTIVE_STATE   0
#define LEDS_INV_MASK       LEDS_MASK

#define LED_START           16
#define LED_1               16
#define LED_2               18
#define LED_STOP            18

#define LEDS_LIST           { LED_1, LED_2 }
#define BSP_LED_0           LED_1
#define BSP_LED_1           LED_2

// Buttons definitions
#define BUTTONS_NUMBER      0
#define BUTTONS_ACTIVE_STATE 0

// Motor definition
#define MOTOR_PIN           20
#define MOTOR_ACTIVE_STATE  1
```

5.7 PCB design

The motivation behind custom PCB design was the integration with wireless power transfer solution from [29] with WBAN multiprotocol application. For this thesis, the main purpose of this PCB is functionality tests of low-cost nRF52810 chip. Minimalism requirement is neglected since larger components and PCB design with no internal copper layers ensure easy assembly and low price.

Connector for monopole antenna was added because the development board nRF52 *DK* does not offer such a feature. Standard 10-pin *Cortex Debug Connector* for SoC programming was added. Optional external crystal X1 and DC/DC converter circuitries were added on the prototype board to evaluate their significance in this solution. Fig. 5.5 and 5.6 show both *Device* and *Host* prototype hardware.

Prototype board design operating in the RF environment has to follow certain rules to maintain proper performance on communication channels. Board is composed of two outside copper layers; it is a minimum number required for this circuitry. Passive components are mostly in *0805* footprints to ensure an easy soldering process. Antenna matching circuitry is in *0402* footprints to avoid parasitic capacitance at the input to the SoC. External oscillators are placed as close as possible to the SoC, they are prioritized upon other discrete components. Switching signals from wireless power and haptic feedback motor are placed as far as possible from the chip, to avoid radio interference.

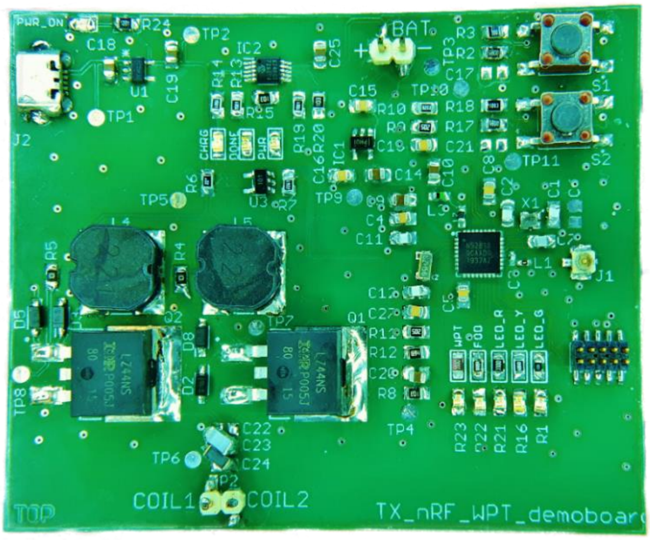

Fig. 5.5: PCB prototype - *Host* side

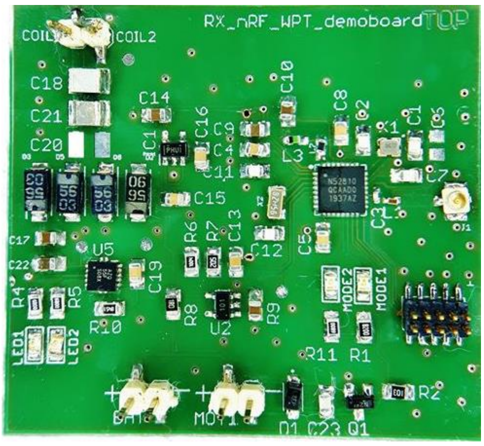

Fig. 5.6: PCB prototype - *Device* side

Chapter 6
System testing

After the assembly process and first successful test runs of the designed board, performance testing of the system can be done. This chapter is about discussing methods used on custom PCB and nRF52 *DK* testing with a focus on individual peripherals and states connected with normal operation. Energy consumption is estimated from measured power profiles in various states of connection. The rating of custom antenna solution is done by comparison of signal transmission performance between development and custom board. Experimental tissue attenuation measurement is done to verify body implant topology. Outputs from this chapter are as independent as possible from external peripherals so that these results can be suitable for systems other than the WBAN node.

6.1 Debug printouts on terminal

Debugging by using breakpoints in BLE applications is not suitable due to time-dependent events in SoftDevice. When attempting to step-by-step debugging through code by breakpoints, application immediately hard faults. This is the reason why printing the observed variables on the terminal is a very useful feature. Multiple informative strings were being printed during software design, for example, user data packets in Gazell protocol communication or voltage measured by SAADC during precision tests. The experiment described in chapter 6.4 used the logging feature for RSSI measurements.

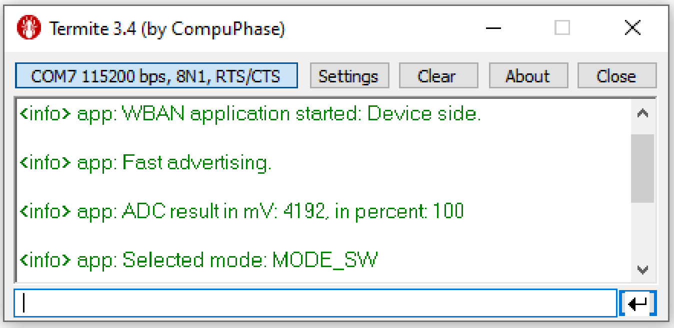

Fig. 6.1: Terminal printouts

Nordic SDK contains the NRF_LOG module that allows terminal printouts by virtual COM port using *nrf_log_info* function. Connection with computer is made by a suitable programmer, nRF52 *DK*'s onboard J-link is sufficient. UART based logging allows users to use RS232 terminals like *Termite*. Another possibility is to use the solution from Segger, Real-Time Terminal. This terminal window is only accessible in debug mode that is not an effective method in certain situations.

To allow logging functionality, module UART_BACKEND needs to be allowed at the sdk_config header file. It is also necessary to allow the NRF_LOG module, as shown in the following code snippet.

```
// <e> NRF_LOG_BACKEND_UART_ENABLED - nrf_log_backend_uart - Log UART backend
//==========================================================
#ifndef NRF_LOG_BACKEND_UART_ENABLED
#define NRF_LOG_BACKEND_UART_ENABLED 1
#endif

// <e> NRF_LOG_ENABLED - nrf_log - Logger
//==========================================================
#ifndef NRF_LOG_ENABLED
#define NRF_LOG_ENABLED 1
#endif
```

6.2 Board support package

This module is useful for indicating the actual states of BLE communication without additional software support. Following structure shows states that are indicated in WBAN application Timeslot indication is added in addition to these states, but it is not part of structure inside the BSP module. Table 4.1 shows how each indication is displayed on the custom PCB.

```
typedef enum
{
    BSP_INDICATE_IDLE,
    BSP_INDICATE_ADVERTISING,
    BSP_INDICATE_CONNECTED,
    BSP_INDICATE_SENT_OK,
} bsp_indication_wban_t;
```

The indication is realized by two onboard LEDs. BSP indication states are displayed only in *WBAN Mode* MODE_READY as described in table 4.1. In the remaining WBAN modes is BSP indication inactive, that is simply done by calling function bsp_indication_set with argument BSP_INDICATE_IDLE. This command shuts off both LEDs so that they can be used for different purposes.

6.3 ADC precision

The precision of ADC signal conversion was measured to discovery and correction of possible interference from RF noise. Measurement was done on custom PCB powered with an adjustable power supply. All communication and peripherals were disabled during the measurement to avoid additional noise from the current draw. The voltage level of the supply was measured both by oscilloscope and by SAADC. The digital value was transferred back into millivolts by equation 4.2. Calculated voltage level was regularly printed on the terminal. Five samples were saved for every voltage level. Mean value was calculated and compared with the voltage measured by the oscilloscope. This process was repeated for five voltage levels linearly spaced inside the working range U_{BAT} = (2.8; 4.2) V. Error was calculated by eq. 6.1. The dependence of error on input voltage is displayed in fig. 6.2.

$$U_{ERR} = \sqrt{(U_{ADC} - U_{OSC})^2} \tag{6.1}$$

Error is not constant over the operation range of battery, but there is an offset caused by voltage drop on resistors that are part of the voltage divider. This offset can be corrected by software inside ADC_RESULT_MV macro.

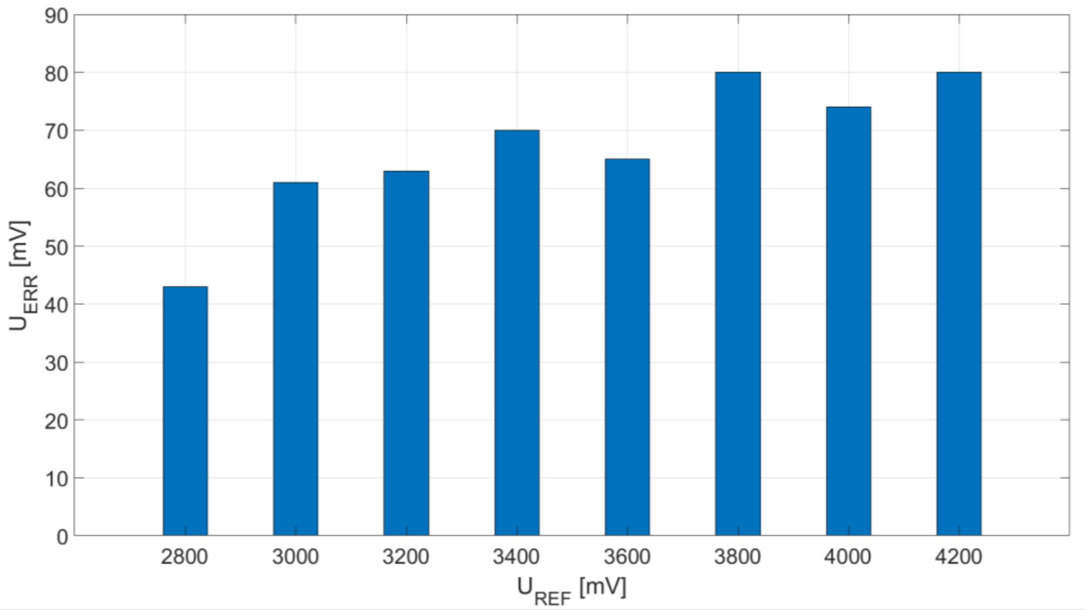

Fig. 6.2: ADC precision measurement

6.4 Connection aspects

To ensure the stable operation of devices, all combinations of connection events that can happen during normal operation have to be tested. Rules are specified and have to be followed during connection events.

- Connection made on one communication protocol is independent of the other one.
- Connection event is performed no matter which device is turned on first and last.
- BLE starts advertising immediately after the startup of *Device*
- BLE starts advertising after BLE disconnect event

Another aspect of connection is possible interference between two devices with the indentical flashed software. This is solved by bonding the dedicated *Host* controller to one certain *Device*. This feature can be done by SoC identification by factory preset parameters or by Gazell Pairing Library.

6.5 Power consumption

Nordic Semiconductor offers an online power profiler for consumption estimation, that is based on the model of the BLE device. However, this profiler does not include proprietary protocols alone nor the Timeslot API. To solve this, the real consumption profile of the *Device* was measured and processed. This measurement was done on the development board since it has brought out pins for current measurement. Voltage drop on 10 Ω resistor connected in series with the load was measured on the oscilloscope. Dealing with low-power, high-frequency signals with standard oscilloscope probe end up with noisy results. Instead, ground needle tip similar to the one in fig. 6.3 was used. Measurements with this fix provided much clearer profiles.

Fig. 6.3: Oscilloscope probe with the ground needle tip

Peripherals were disabled during this measurement because their current draw interfered with radio activity. Development board was powered by adjustable DC power supply, U_{IN} = 3,3 V. Both BLE and Gazell transmitted at maximum power P_{TX} = +4 dBm. Function for Gazell data transmission was called every 120 ms. The advertising interval was 187,5 ms. Measured data were exported into a .dat file and processed in *Matlab*. The current profile was calculated using Ohm's law. Average current values were obtained from measured time-domain signals according to eq. 6.2. Cumulative trapezoidal numerical integration was used to obtain average current values from measured time-domain signals.

$$I_{AVG} = \frac{1}{T}\int_0^T i(t)dt \tag{6.2}$$

The following graphs provide the current profiles of two connection states. In fig. 6.4 is the *Device* in advertising mode and there is no *Host* connected by Gazell link. Two advertising events ($t_1 \approx 30$ ms & $t_2 \approx 210$ms) and three timeslots can be observed. Inside timeslot, Gazell is repeatedly trying to establish the connection with *Host*. The average current value is in this case $I_{ADV} = 3.346$ mA.

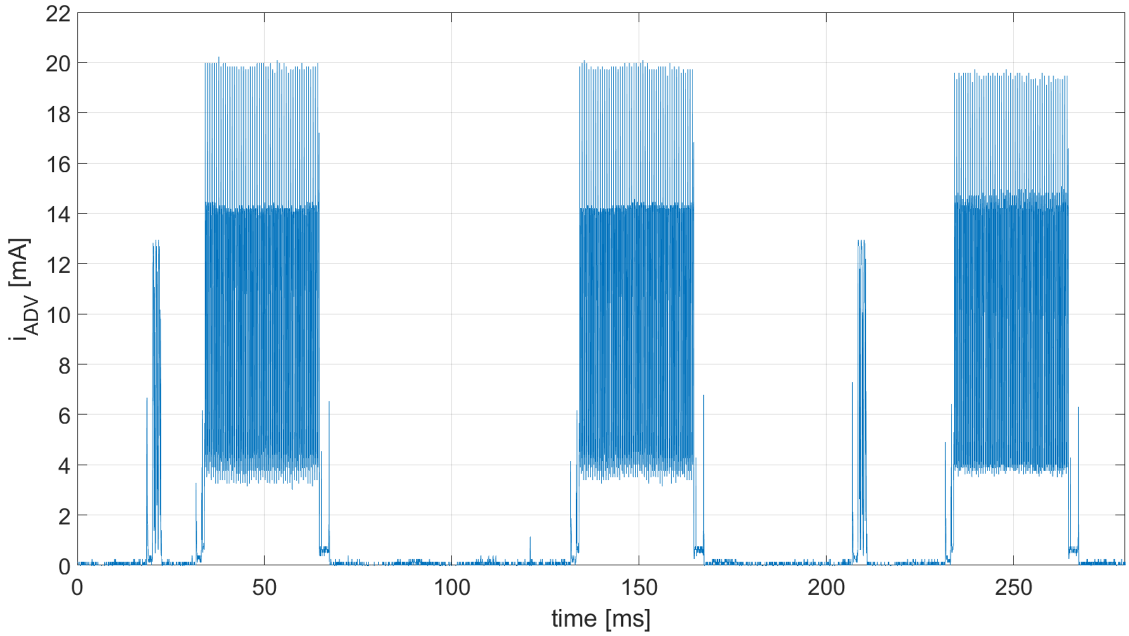

Fig. 6.4: Current consumption profile in the disconnected state

The second graph in fig. 6.5. shows *Device* in state of connection both by BLE and Gazell. Two connection events and three timeslots are visible again, with timeslot consumption visibly decreased. The average value confirms statement, $I_{CONN} = 1.117$ mA.

Measured current profiles prove that proprietary protocol demands much more energy compared with BLE, mainly in the disconnected state. There are methods that minimize Gazell consumption. Transmitted power has a significant impact on consumption, and it is not always necessary to use the highest possible one. Connection attempts can be limited by setting the *nrf_gzll_set_max_tx_attemts* function. The data exchange period in the connected state can be also extended as well as timeslot duration and period. On the other hand, these actions can cause communication delays, so certain level of optimization is required.

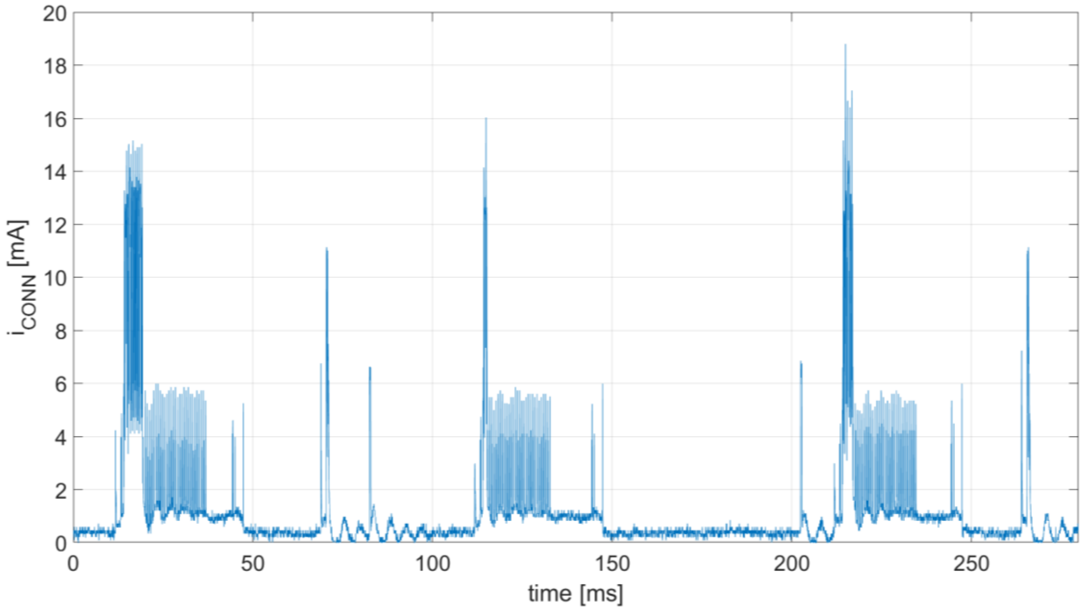

Fig. 6.5: Current consumption profile in the connected state

6.6 Range measurements

The purpose of this test is custom monopole antenna performance comparison with the development board's properly tuned PCB antenna in real-life indoor conditions. The range was measured in RSSI units described in chapter 3.3. Identical WBAN firmware was loaded both to development and custom boards. RSSI was measured with features provided in the *Lightblue* smartphone application.

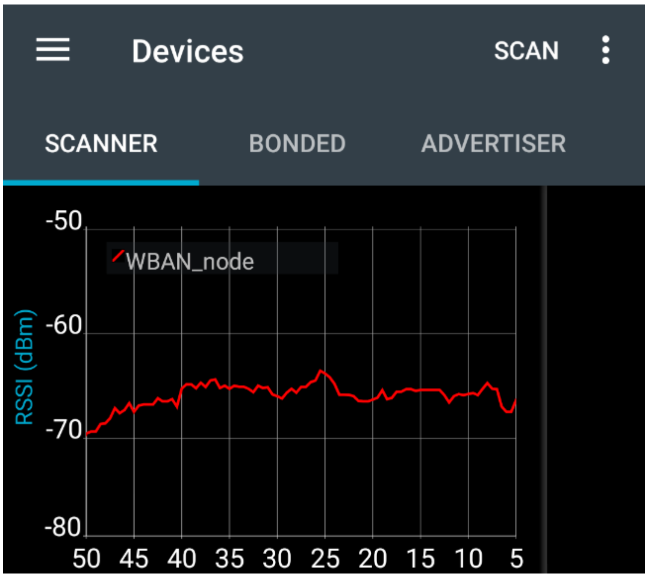

Fig. 6.6: RSSI measurement observed from the smartphone

Five values for each of the five different distances between transmitter and receiver were obtained. Fig. 6.7 shows the difference between data measured on each of the boards. Antenna on the development board has slightly better performance in the middle of measured range, however, results are satisfactory considering the custom antenna simplicity.

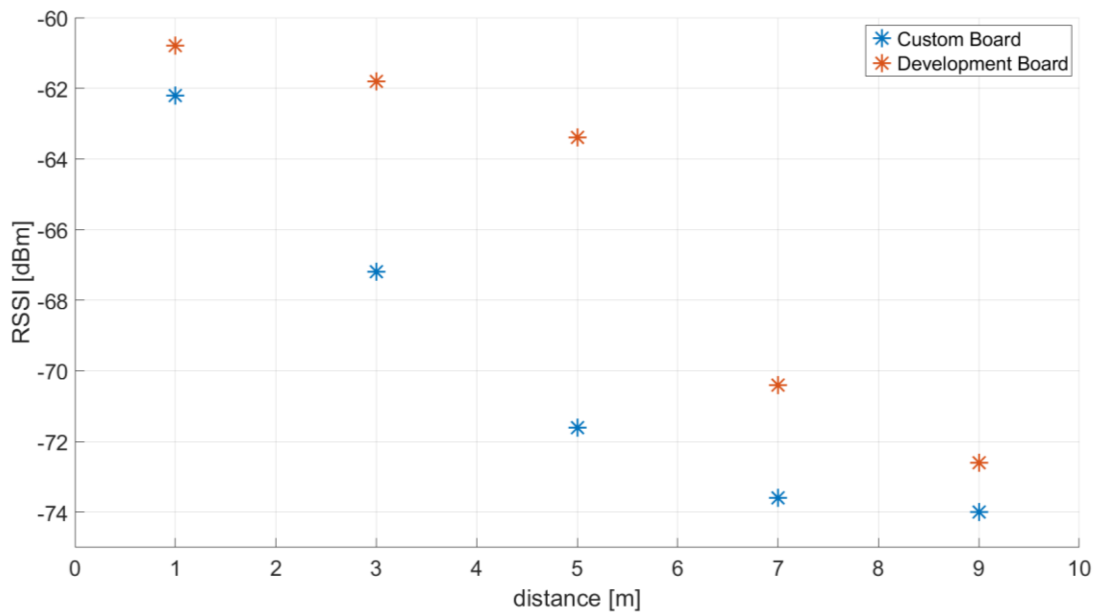

Fig. 6.7: RSSI measurement comparison

6.6.1 Tissue attenuation experiment

The main task of the experiment described in this chapter was verification of signal attenuation caused by the human body, considering the topology from Chapter 2.3. In case of the body implant, the entire *Device* might be fully covered in tissue, so the body-shadowing effect is worth observing. Scenarios with PCB and monopole antenna from coaxial cable were evaluated. There are two ways how to reconstruct implant topology. First, thanks to the small factor of custom PCB, is coverage of the entire device with hands, however, this solution does not ensure the sufficient coverage. The other possibility is to use animal tissue, in this case, ground beef. Fig. 6.9 documents how the experiment was performed. Custom PCB was wrapped into the tissue and served as a transmitter. RSSI values were measured by smartphone, as described in previous chapters. RSSI was measured for multiple distances from the transmitter, similar to those that are expected as common operation distance.

First, transmitter with PCB antenna was tested. In this attempt, attenuation was so significant in this topology that it was not possible to establish the connection, despite full transmitter power. This caused that RSSI could not be measured, that is why it is not included in the results. Second tested device was the custom *Device* board with exposed monopole antenna. This attempt was more successful, RSSI was measured and graph from fig. 6.8 presents the comparison between the fully covered *Device* with exposed antenna and a free-in-space state, considering same antenna orientation. Significant attenuation of implant topology can be observed at 9 m distance from the transmitter. This is caused by the absorption of a certain amount of radiation since the antenna was placed above the tissue. Despite this fact, monopole antenna made from coaxial cable solves the problem of body attenuation with satisfactory results.

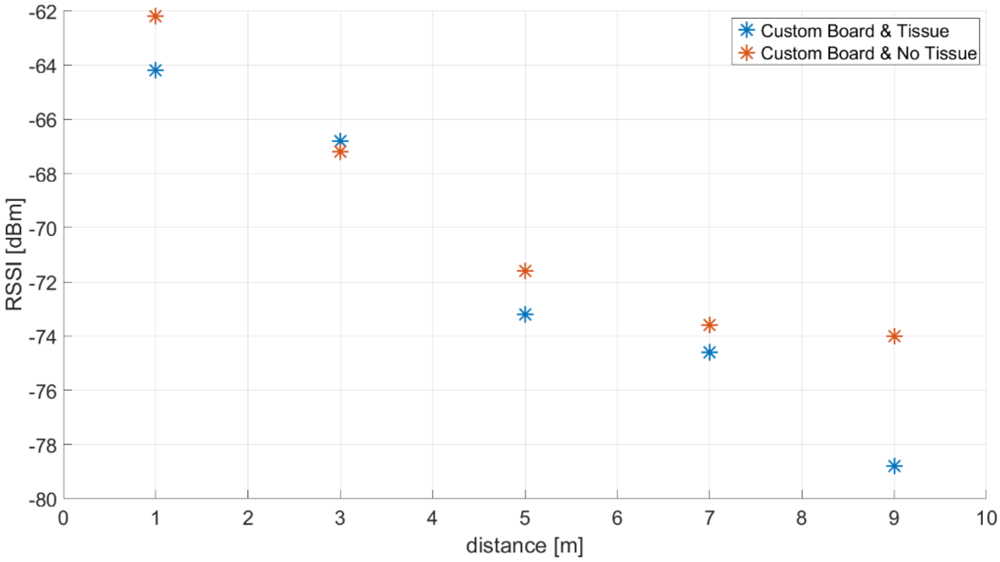

Fig. 6.8: RSSI values from tissue experiment

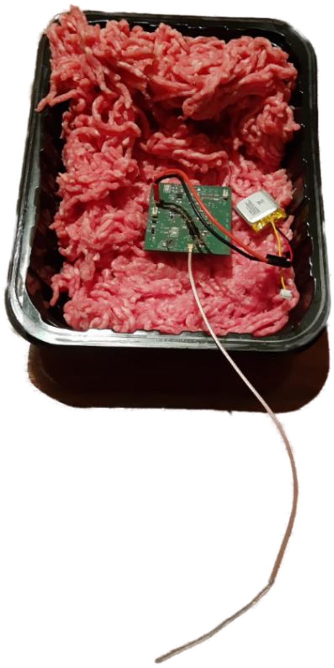

Fig. 6.9: Tissue experiment

Chapter 7
Conclusion and future work

The goal of this thesis, multiprotocol functionality of battery-operated device, was implemented on suitable hardware, nRF52810 SoC. Hardware selection process was based on pre-selected criteria and recherche of problematic to suit application demands.

Software design was created in Segger Embedded Studio, debugged on nRF52 *DK* with selected hardware emulation. Bluetooth Low Energy protocol stack was implemented with custom service for *WBAN Mode* selection. Native Bluetooth battery Service is included, these two attributes provide smartphone access into the application. Proprietary protocol named Gazell ensures communication between multiprotocol *Device* and dedicated remote controller. Application data exchange similar to BLE can be observed, although with certain amount of simplification in terms of robustness. Both protocols share the same radio peripherals, including custom pigtail monopole antenna with U.FL RF connector.

Battery voltage measurement with successive-approximation ADC is the transmitted parameter in direction from multiprotocol *Device*. State of charge can be observed either by LED indication on dedicated *Host* or by characteristic value of battery service. Data transferred into WBAN *Device* contain command to set certain mode, presented by multiple indication states with the use of two on-board LEDs and small vibration motor. Purpose of this motor is haptic feedback reminding that the *Device* should be charged.

Custom PCB design includes battery management solution, vibration motor driver circuit and reference layout of wireless SoC. Rules required for proper wireless data transmission range are followed. Prototype result is verified by series of tests exploring range performance by RSSI units, consumption profiles and ADC precision measurement. Measured data are processed in Matlab.

Possible implant topology of WBAN *Device* led to an experiment that verified signal attenuation by human tissue. Prototype with covered PCB antenna was not able to establish nor maintain the connection. Solution with custom antenna reported satisfactory results with minimal attenuation in typical indoor distances.

Firmware from this thesis is prepared for front-end smartphone application design. Schematics can be used for PCB design using components with smaller footprints, according to the given size requirements

Bibliography

[1] GEORGAKAKIS, E., S.A. NIKOLIDAKIS, D.D. VERGADOS and C. DOULIGERIS. An Analysis of Bluetooth, Zigbee and Bluetooth Low Energy and Their Use in WBANs. *Wireless Mobile Communication and Healthcare: MobiHealth* [online]. 2011, **2010**(55), 168-175 [cit. 2020-06-18]. Available from: <https://link.springer.com/chapter/10.1007/978-3-642-20865-2_22>.

[2] LONG, M. *Wireless Audio Explained: Bluetooth vs 2.4GHz Audio* [online]. 2019-07-04 [cit. 2020-06-18]. Available from: <https://www.electropages.com/blog/2019/07/wireless-audio-explained-bluetooth-vs-24ghz-audio>.

[3] AREFIN, T., FAZLUL, A.K.M, HAQUE, HANIF, M. Wireless Body Area Network: An Overview and Various Applications. *Journal of Computer and Communications* [online]. **2017**(5), 53-64 [cit. 2020-06-18]. Available from: <https://www.researchgate.net/publication/316989520_Wireless_Body_Area_Network_An_Overview_and_Various_Applications>.

[4] MEDTRONIC. *Azure Pacing System for Bradyarrhythmia Management* [online]. 2017 [cit. 2020-06-18]. Available from: <https://www.medtronic.com/us-en/healthcare-professionals/products/cardiac-rhythm/pacemakers/azure.html>.

[5] DOLEJŠ, J. *Proč je síla signálu telefonu uváděna v záporných hodnotách a co vyjadřuje?* [online]. 2015-03-18 [cit. 2020-06-18]. Available from: <https://www.svetandroida.cz/sila-signalu/>.

[6] GAO, V. *Proximity and RSSI* [online]. 2015-09-21 [cit. 2020-06-18]. Available from: <https://www.bluetooth.com/blog/proximity-and-rssi/>.

[7] NXP SEMICONDUCTORS. *BLE Antenna Design Guide* [online]. 2013, Revised on: 2018-05-09 [cit. 2020-06-18]. Available from: <https://www.nxp.com/docs/en/user-guide/UM10992.pdf>.

[8] PATTNAYAK, T., THANIKACHALAM, G. *Antenna Design and RF Layout Guidelines* [online]. 2014 Revised on: 2018-08-30 [cit. 2020-06-18]. Available from: <https://www.cypress.com/file/136236/download>.

[9] *The Antenna Theory Website* [online]. 2009 [cit. 2020-06-18]. Available from: <http://www.antenna-theory.com>.

[10] *WLAN WiFi Bluetooth Small Rubber Antenna* [online]. In: . [cit. 2020-06-19]. Available from: <https://tekfun.en.taiwantrade.com/product/11mm-2-4ghz-wlan-wifi-bluetooth-small-rubber-antenna-with-pigtail-cable-and-ipex-sma-flying-leads-1724589.html>.

[11] NORDIC SEMICONDUCTOR. *Antenna tuning* [online]. 2012 [cit. 2020-06-19]. Available from: https://infocenter.nordicsemi.com/pdf/nwp_017.pdf

[12] SCHOLZ, N. *Mobile phones and health: Where do we stand?* [online]. 2019 [cit. 2020-06-19]. Available from: <https://www.europarl.europa.eu/RegData/etudes/BRIE/2019/635598/EPRS_BRI(2019)635598_EN.pdf>.

[13] KURUP, D., JOSEPH, W., VERMEEREN, G., MARTENS, L. *Path loss model for in-body communication in homogeneous human muscle tissue* in *Electronics Letters*, vol. 45, no. 9, pp. 453-454, 23 April 2009, [cit. 2020-06-19]. doi: 10.1049/el.2009.3484.

[14] RUSH A.D., TROYK P.R. *A Power and Data Link for a Wireless-Implanted Neural Recording System* in *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 11, pp. 3255-3262, Nov. 2012, [cit. 2020-06-19]. doi: 10.1109/TBME.2012.2214385.

[15] PRECISION MICRODRIVES. *Linear Resonant Actuators – LRAs* [online]. [cit. 2020-06-19]. Available from: <https://www.precisionmicrodrives.com/vibration-motors/linear-resonant-actuators-lras/>.

[16] OFCOM. *Frequency bands designated for Industrial, Scientific and Medical use (ISM)* [online]. 2017 [cit. 2020-06-19]. Available from: <https://www.ofcom.org.uk/__data/assets/pdf_file/0022/103297/fat-ism-frequencies.pdf>.

[17] MATHWORKS. *BLE Channel Selection Algorithms* [online]. [cit. 2020-06-19]. Available from: <https://www.mathworks.com/help/comm/examples/ble-channel-selection-algorithms.html>.

[18] BUI, Hung. *Bluetooth Smart and the Nordic's Softdevices: Part 2 Connection* [online]. 2016-02-01 [cit. 2020-06-18]. Available from: <https://devzone.nordicsemi.com/nordic/short-range-guides/b/bluetooth-low-energy/posts/bluetooth-smart-and-the-nordics-softdevices-part-2>.

[19] *BLE Advertising and Scanning* [online]. In: . [cit. 2020-06-19]. Available from: <https://github.com/qdfreqchip/FR801x-SDK/wiki/BLE-Advertising-and-Scanning>.

[20] BLUETOOTH SIG. *Bluetooth Generic Attributes* [online]. 2020 [cit. 2020-06-18]. Available from: <https://www.bluetooth.com/specifications/gatt/>.

[21] TEXAS INSTRUMENTS INCORPORATED. *SimpleLinkTM multiprotocol wireless MCUs* [online]. 2019 [cit. 2020-06-19]. Available from: <https://www.ti.com/wireless-connectivity/simplelink-solutions/multi-standard/overview.html>.

[22] SILICON LABS. *Maintaining Flexibility with Multiprotocol Wireless Connectivity* [online]. 2018 [cit. 2020-06-19]. Available from: <https://www.silabs.com/wireless/multiprotocol>.

[23] NORDIC SEMICONDUCTOR. *NRF52810 Product Specification v1.3.* [online]. [cit. 2020-06-18]. Available from: <https://infocenter.nordicsemi.com/pdf/nRF52810_PS_v1.3.pdf>.

[24] ON SEMICONDUCTOR. *RSL10 Datasheet* [online]. 2020 [cit. 2020-06-18]. Available from: <https://www.onsemi.com/pub/Collateral/RSL10-D.PDF>.

[25] STMicroelectronics. *BlueNRG-M2 Datasheet* [online]. 2019, Revised on: 2019-07-24 [cit. 2020-06-18]. Available from: <https://www.st.com/resource/en/datasheet/bluenrg-m2.pdf>.

[26] NORDIC SEMICONDUCTOR. *S112 SoftDevice Specification* [online]. 2017, Revised on: 2019-09-01 [cit. 2020-06-18]. Available from: <https://infocenter.nordicsemi.com/pdf/S112_SDS_v3.1.pdf/>.

[27] SEGGER MICROCONTROLER SYSTEMS. *J-Link OB* [online]. [cit. 2020-06-19]. Available from: <https://www.segger.com/products/debug-probes/j-link/models/j-link-ob/>.

[28] NORDIC SEMICONDUCTOR. *NRF5 SDK v16.0.0 Release notes* [online]. 2019 [cit. 2020-06-19]. Available from: <https://infocenter.nordicsemi.com/index.jsp?topic=%2Fsdk_nrf5_v16.0.0%2Findex.html&cp=7_1_0>.

[29] BEDNAŘÍK, Josef. *Návrh a realizace bezdrátového nabíjení pro vestavěné systémy*. Brno, 2020. Available from: <https://www.vutbr.cz/studenti/zav-prace/detail/124655>. Master’s thesis. Brno University of Technology, Faculty of mechanical engineering, Institute of solid mechanics, mechatronics and biomechanics. Supervised by doc. Ing. Jiří Krejsa, Ph.D.

[30] *8mA current draw with timeslot API, BLE + Gazell host* [online]. 2019 [cit. 2020-06-18]. Available from: <https://devzone.nordicsemi.com/f/nordic-q-a/42456/8ma-current-draw-with-timeslot-api-ble-gazell-host-softdevice-130>.

[31] *Online GUID Generator* [online]. [cit. 2020-06-19]. Available from: <https://www.guidgenerator.com/online-guid-generator.aspx>.

[32] PAUL. *Battery selection for IoT design* [online]. 2019 [cit. 2020-06-19]. Available from: <https://www.disk91.com/2019/technology/hardware/battery-selection-for-iot-design/>.

[33] TEXAS INSTRUMENTS INCORPORATED. *bq24012, Single-Chip, Li-Ion Charge Management IC Datasheet* [online]. 2002 [cit. 10-5-2020]. Available from: <http://www.ti.com/lit/ds/symlink/bq24012.pdf>.

[34] AFANEH, M. *How to build the simplest nRF52 BLE Peripheral application (Lightbulb use case)* [online]. 2018-05-29 [cit. 2020-06-24]. Available from: <https://www.novelbits.io/smart-ble-lightbulb-application-nrf52/>.

List of abbreviations

API	Application Programming Interface
ATT	Attribute
BLE	Bluetooth Low Energy
CC/CV	Constant Current/ Constant Voltage
ESB	Enhanced Shock Burst
GAP	Generic Access Profile
GATT	Generic Attribute
GCC	GNU Compiler Collection
GPIO	General Purpose Input Output
IDE	Integrated Development Environment
IoT	Internet of Things
IRQ	Interrupt Request
ISM	Industrial Scientific Medical
L2CAP	Logical Link Control and Adaption Protocol
Q&A	Questions and Answers
RF	Radio Frequency
RSSI	Received Signal Strength Indication
RTC	Real Time Clock
SAADC	Successive Approximation Analog to Digital Converter
SAR	Specific Absorption Rate

SDK	Software Development Kit
SIG	Special Interest Group
SiP	System in Package
SoC	System on Chip
SVC	Super Visor Call
SWD	Serial Wire Debug
UART	Universal Asynchronous Receiver Transmitter
UHF	Ultra High Frequency
UUID	Universally Unique Identifier
WBAN	Wireless Body Area Network

Contents of enclosed archive

Archive WBAN_2020_archive contains following folders:

wban_device_fw	Firmware for *Device* with structure described in Chapter 4
wban_device_pcb	PCB layout of *Device*
wban_host_fw	Firmware for dedicated *Host*
wban_host_pcb	PCB layout of *Host*

To ensure correct functionality of firmware, archive must be extracted to the *projects* folder inside *nRF5_SDK_16.0.0*, following path specified below:

C:\...\nRF5_SDK_16.0.0\worskpace\projects\WBAN_2020_archive